Aalto University
School of Science
Master's Programme in Computer, Communication and Information Sciences

Siddharth Ramchandran

Latent Gaussian processes with composite likelihoods for data-driven disease stratification

Master's Thesis
Espoo, July 10, 2019

Supervisor: Professor Harri Lähdesmäki, Aalto University, Finland
Advisor: Miika Koskinen, HUS Helsinki University Hospital, Finland

Aalto University
School of Science
Master's Programme in Computer, Communication and Information Sciences

ABSTRACT OF MASTER'S THESIS

Author: Siddharth Ramchandran

Title:
Latent Gaussian processes with composite likelihoods for data-driven disease stratification

Date:	July 10, 2019	**Pages:**	71
Major:	Computer Science	**Code:**	SCI3042
Supervisor:	Professor Harri Lähdesmäki, D.Sc.		
Advisor:	Miika Koskinen, D.Sc.		

Machine learning has caused a seismic shift on how clinical patient data is being used and interpreted. It can be harnessed for more effective and efficient healthcare that can benefit both patients and medical practitioners through personalised health solutions. Disease stratification is an important task in personalised medicine and has the potential to help medical researchers better understand diseases. In collaboration with the Helsinki Biobank and the Helsinki University Hospital, we aim to better understand clinical patient records comprising of multiple likelihoods (with noisy and missing values) by embedding these high-dimensional observations in to a low-dimensional space while capturing the similarity between the observations.

In this thesis, we propose an unsupervised, generative model that can identify this latent clustering among patients while making use of all available data (i.e., in a heterogeneous data setting). We make use of deep neural networks and Gaussian process latent variable models (GPLVM) to create a form of non-linear dimensionality reduction for heterogeneous data.

The key principle in our model is to use the output of latent GPs (sparse GPs) to modulate the parameters of the different likelihoods through link functions. The intractability introduced by the composite likelihoods is overcome by making use of sampling-based variational inference with quadrature. We make use of deep neural networks to parameterise the variational inference to introduce a constraint that balances between locality and dissimilarity preservation in the latent space.

We demonstrated the effectiveness of our model on toy datasets and clinical data of Parkinson's disease patients treated at the HUS Helsinki University Hospital. Our approach identifies sub-groups from the heterogeneous patient data and we evaluated the differences in characteristics among the identified clusters using standard statistical tests.

Keywords: GPLVM, sparse GP, neural network, variational inference, personalised medicine

Language: English

Acknowledgements

Throughout my Master's studies, I have received a great deal of support and advice from many people. I would like to express my sincere gratitude to Prof. Harri Lähdesmäki for his constant encouragement and guidance. His knowledge and insights have been truly invaluable to my work. I am equally grateful for the support, advice and time of Dr. Miika Koskinen. His efforts and help were crucial to this project.

I would like to thank Gleb Tikhonov, Henrik Mannerström and Charles Gadd for their suggestions and feedback on this project. Also, I would like to thank Juhi Somani and Charles Gadd for their encouragement and camaraderie.

Finally, I would like to thank Pradeep Eranti and Parvati Pillai for always being there for me. Last, but definitely not the least, I am forever grateful to my parents for their love and patience.

Espoo (Finland) - July 10, 2019

Siddharth Ramchandran

We would like to acknowledge the computational resources provided by the Aalto Science-IT, Finland.

This work has resulted in a paper, 'Ramchandran, Koskinen, Lähdesmäki (2019). *Latent Gaussian processes with composite likelihoods for data-driven disease stratification*'

Abbreviations and Acronyms

PCA	Principal component analysis
PPCA	Probabilistic principal component analysis
GP	3rd Gaussian process
GPLVM	Gaussian process latent variable model
LLE	Locally linear embedding
GPU	Graphical processing unit
KL	Kullback-Leibler divergence
ELBO	Evidence lower bound
ARD RBF	Automatic relevance determination radial basis function
MLP	Multi-layer perceptron
ANN	Artificial neural network
RELU	Rectified linear unit
SGD	Stochastic gradient descent
ADAM	Adaptive moment estimation
GMM	Gaussian mixture model
EM	Expectation-Maximisation
BIC	Bayesian information criterion
ICD	International classification of disease codes

Contents

Chapter 1

Introduction

Disease stratification plays an important role in understanding diseases that exhibit a high degree of heterogeneity such as diabetes and Parkinson's disease. Moreover, disease stratification is used in current personalised medicine approaches that promise targeted prediction, prevention and treatment of diseases [Achenbach et al., 2004; Harvey et al., 2012]. Disease stratification refers to the identification of underlying sub-groups or a latent structure within a cohort of clinical data. Cluster analysis can play a vital role in detecting disease heterogeneity and identifying these sub-groups. It can be defined as the categorisation of similar objects (or clinical observations related to patients in our case) into groups, where the number of groups is usually unknown [Kaufman and Rousseeuw, 2009]. For example, Ahlqvist et al. [2018] made use of k-means and hierarchical clustering techniques on a diabetes cohort to find clusters of patients that express significantly different characteristics and risk of diabetic complications among the clusters.

Another approach to achieve disease stratification would be to obtain a low-dimensional representation which reveals some latent structure in the data. For high-dimensional clinical data it is also vital to have a suitable low-dimensional representation which can be visualised for further analysis. Moreover, clinical data or patient records usually comprise of high-dimensional data from several disparate sources. In a statistical setting, these disparate sources or observation spaces are represented by different likelihoods and may correspond to disease codes, laboratory measurements, disease symptoms, etc.

Principle component analysis (PCA) is perhaps one of the most popular techniques for dimensionality reduction. It can be motivated as seeking an orthogonal linear projection of the data along the direction of maximum vari-

ance (i.e., projecting the data along its principal components) [Jolliffe, 2011]. PCA can be seen as a linear approach to dimensionality reduction. Lawrence [2004] reinterpreted PCA as a Gaussian process (GP) mapping from a latent space to data space and proposed a generalisation by using a prior that allows for non-linear processes called Gaussian process latent variable model (GPLVM). In short, the GPLVM attempts to learn a smooth mapping from latent space to data space.

To accurately capture the latent manifold structure of the data, it is important for a dimensionality reduction algorithm to balance between preserving the distance between nearby data points and ensuring that data points that are distant in the data space are not nearby in the latent space (dissimilarity). However, the GPLVM algorithm only guarantees the later and does not have any constraint that ensures the former. Lawrence and Quiñonero-Candela [2006] discusses this issue in detail and introduces the idea of incorporating a local-distance preserving constraint thereby formulating a back-constrained GPLVM. We impose this constraint using recognition models (or neural networks) which introduces a mapping from data space to latent space [Bui and Turner, 2015]. The recognition models also allows the introduction of efficient mini-batching to the optimisation of the GPLVM. To summarise, we have two models: the recognition model that preserves local distances and the GPLVM that preserves the dissimilarities.

GPLVMs are targeted towards homogeneous datasets (i.e., data from a single observation space or likelihood). This poses a significant challenge in our setting in which we have clinical data which can be described by multiple likelihoods. We build upon the idea of obtaining a shared latent space or a common low-dimensional latent representation using a shared GPLVM as proposed in Ek et al. [2007]. In other words, we extend the idea of shared GPLVM to support multiple likelihoods. The use of non-Gaussian likelihoods introduces intractability into the inference. Titsias and Lawrence [2010] introduced variational inference to GPLVMs that assume the standard Gaussian noise model. However, this cannot be extended to multiple likelihoods due to the absence of an analytical solution for the optimal variational distribution. We overcome this by using a sampling-based variational inference with quadrature. The faster convergence achieved by using mini-batching with the recognition model compensates for the sampling overheads. The introduction of the recognition models brings our method closer to the autoencoder. Autoencoders try to learn a latent representation using a multi-layer neural network to encode the data from the data space to a low-dimensional latent space (encoder) and a separate neural network to decode the data from

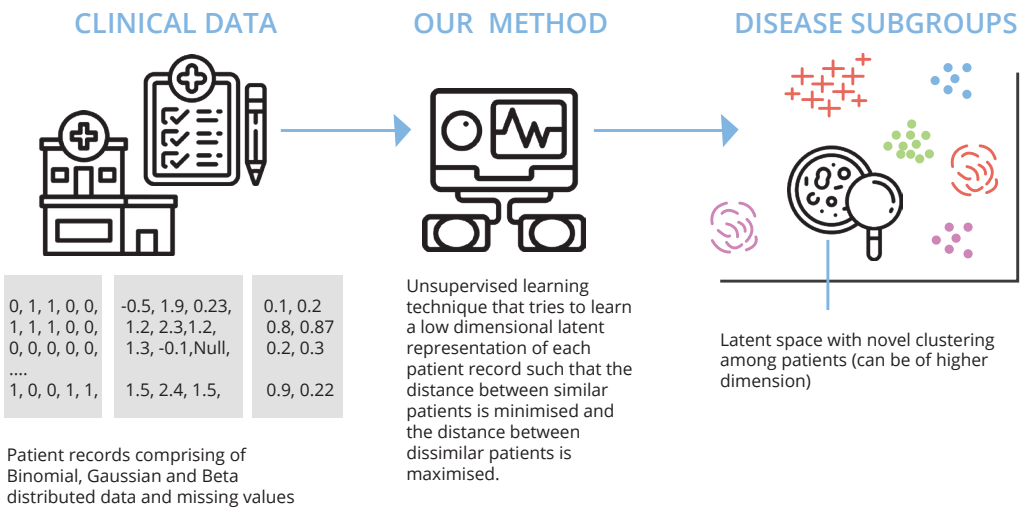


Figure 1.1: Discovering latent clustering from clinical patient records.

the low-dimensional latent space back to the data space (decoder) [Hinton and Salakhutdinov, 2006]. In our approach, the recognition model introduced into the extended GPLVM architecture acts as a form of encoder, while the GPs act as a decoder.

1.1 Problem statement

In this study, we aim to better understand patient records comprising of multiple likelihoods by embedding these high dimensional observations into a low-dimensional space while capturing the similarity between the observations. Patient records may contain noisy data and missing values. Hence, it is vital that our method suitably handles missing values and is robust to outliers. Figure 1.1 visualises our objective.

This thesis proposes an extension of the Gaussian process latent variable models (GPLVMs) to produce low-dimensional embeddings of heterogeneous datasets while preserving the similarities between the observations. In other words, we propose our method as a form of non-linear dimensionality reduction for heterogeneous data. The key principle in our model is to use the outputs of latent Gaussian processes to modulate the parameters of the different likelihoods through the use of link functions. We demonstrate the applicability of our proposed method on clinical data of Parkinson's disease. Our method can be seen as a generalisation of Gal et al. [2015] for heteroge-

neous data while scalable to larger datasets.

1.2 Structure of the Thesis

This thesis is structured into six sections. In Section 2, we survey related works while introducing the concepts and theoretical background required in this thesis. A detailed description of the datasets can also be found in this section. In Section 3, we describe our model in detail with all the relevant derivations. We present the results of our analysis on publicly available datasets and clinical datasets from the Helsinki Biobank in Section 4. Finally, we conclude this thesis with a short summary and possible avenues for future research in Section 5.

Chapter 2

Background

Finding disease subgroups in an unsupervised fashion is an important task in personalised medicine. It has the potential to help medical researchers better understand diseases and even has the potential to enable medical practitioners to prescribe personalised and targeted prescriptions. This thesis aims to create an unsupervised, generative model that can identify this latent clustering among patients while making use of all available data (i.e., in a heterogeneous data setting). Clinical patient records comprise of data from different disparate sources with different likelihoods. This data will also comprise of missing values and noise.

In this chapter, we shall explain some of the relevant concepts, review the existing literature and describe the datasets.

2.1 Gaussian processes

Gaussian processes (GPs) are powerful and flexible models that have wide applicability in machine learning. They can be interpreted as a generalisation of the Gaussian distribution. According to Rasmussen and Williams [2006], GPs can be contemplated as defining a distribution over functions with inference taking place directly in the space of functions. Given a vector of points, $X = (x_1, x_2, \ldots, x_N)$, these non-parametric machine learning models assigns a random variable $f(x_i)$, $\forall x_i$ where $i = 1, \ldots, N$ and thereby assumes that the joint distribution $p(f(x_1), f(x_2), \ldots, f(x_N))$ is jointly Gaussian with mean function $\mu(x)$ and covariance function Σ_x given by a positive semi-definite kernel k, such that $\Sigma_{i,j} = k(x_i, x_j)$ where $x \in \mathcal{X}$ ($\mathcal{X}$ is the domain of x) [Murphy, 2012]. This can be written as $f \sim \mathcal{GP}$. The covariance function helps to regulate the smoothness of the resulting function (through

the length-scale parameter) and ensures that values that are close together in the input space would result in values that are close in output space. A GP can be defined as,

$$p(\boldsymbol{f}|X) = \mathcal{N}(\mu(x), k(x, x')), \tag{2.1}$$

where $\boldsymbol{f} = f(X) = (f(x_1), f(x_2), ..., f(x_N))$. We assume that the mean $\mu(x)$ is zero and $k(x, x')$ has kernel parameters θ, i.e., $k(x, x'|\theta)$. Extending this to D dimensional inputs (multi-dimensional inputs) where $X = (x_1, x_2, \ldots, x_N) \in \mathbb{R}^{N\times D}$, we write the Gaussian prior as, $f(X) \sim \mathcal{N}(0, K_{X,X}(\theta))$ where we define the elements of the covariance matrix by the kernel $[K_{X,X}(\theta)]_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j|\theta)$. In this thesis, we make use of the automatic relevance determination radial basis function covariance function which can be defined as,

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j|\theta) = \sigma^2_{\text{rbf}} \exp\left[-\frac{1}{2}\sum_{d=1}^{D}\left(\frac{x_{i,d} - x_{j,d}}{\ell_d}\right)^2\right],$$

where $\boldsymbol{\ell}_{\text{rbf}} = (\ell_1, \ldots, \ell_D)$ are the length-scale parameters with a separate value for each dimension and σ^2_{rbf} is the signal variance for the kernel. Hence, the kernel parameters are denoted as $\theta = (\boldsymbol{\ell}_{\text{rbf}}, \sigma^2_{\text{rbf}})$. From now on, we shall denote $K_{X,X}(\theta)$ as $K_{X,X}$ for brevity.

In the supervised setting, after having observed y we can use Equation (2.1) to get the GP posterior, $p(\boldsymbol{f}|X, y)$. Moreover, given new data X_* it is possible to make predictions $\boldsymbol{f}_*$,

$$\begin{aligned} p(\boldsymbol{f}_*|X_*, X, y) &= \int p(\boldsymbol{f}_*|X_*, \boldsymbol{f})p(\boldsymbol{f}|X, y)d\boldsymbol{f} \\ &= \mathcal{N}(\boldsymbol{f}_*|\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*). \end{aligned} \tag{2.2}$$

Hence, from Equation (2.2), the posterior predictive distribution is also Gaussian where:

$$\begin{aligned} \boldsymbol{\mu}_* &= K^T_{X,X_*}K^{-1}_y y \\ \boldsymbol{\Sigma}_* &= K_{X_*,X_*} - K^T_{X,X_*}K^{-1}_y K_{X,X_*}, \end{aligned}$$

where $K_y = K_{X,X} + \sigma^2_y I$ and σ^2_y is the noise term. Figure 2.1 visualises the posterior predictive distribution for some noisy toy training data. The dashed lines represent random functions sampled from the predictive posterior while the solid line represents the mean. The shaded area corresponds to the 95% confidence region. The hyper-parameters (θ and σ^2_y) are optimised by maximising the marginal log-likelihood given by:

$$\log p(y|X) = \log \mathcal{N}(y|0, K_y). \tag{2.3}$$

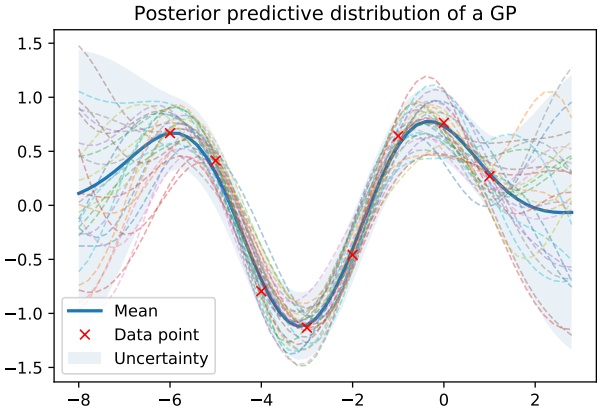


Figure 2.1: Visualisation of posterior predictive distribution given some toy training data.

The application of GPs to supervised learning tasks is quite straightforward. However, the application of GPs to unsupervised learning problems (as is done in this thesis) can be quite involved.

2.2 Dimensionality reduction

Dimensionality reduction is the process of reducing the number of variables under consideration by obtaining a set of principal variables [Roweis and Saul, 2000]. This can involve either feature selection or feature projection. Most techniques focus on preserving the distances between nearby objects than objects that are further apart (i.e. local distance preservation). However, it is also important to ensure that objects that are further apart in data space are also kept apart in the reduced space (i.e. dissimilarity preservation). Unfortunately, in most cases it is not possible to achieve both.

In this work, we focus on feature projection which is the transformation of data from a high-dimensional space to lower dimensional manifold. There are many popular algorithms for dimensionality reduction such as PCA [Jolliffe, 2011], kernel PCA [Schölkopf et al., 1997], locally linear embedding [Roweis and Saul, 2000], Isomap [Tenenbaum et al., 2000], GPLVM [Lawrence, 2004], etc. These algorithms are suitable in the homogeneous data setting and do not necessarily extend well to a heterogeneous data setting. We build upon the GPLVM to support a heterogeneous data setting.

2.2.1 Principle Component Analysis

Principle Component Analysis (PCA) is one of the most popular techniques for linear dimensionality reduction. It tries to identify an orthogonal linear projection of the data along the direction of maximum variance. In other words, it projects a number of possibly correlated variables into a smaller number of uncorrelated variables. These uncorrelated variables are known as the principle components. [Jolliffe, 2011]

PCA involves transforming the original data in such a way that the transformed independent variables are in the decreasing order of independence. It involves finding the eigenvectors and eigenvalues of the covariance matrix of the mean-centred or standardised data. After obtaining the eigendecomposition of the covariance matrix, the transformed data is obtained by multiplying the original data with the eigenvectors sorted in the decreasing order of their corresponding eigenvalues. As an example, consider a D-dimensional dataset Y. The eigendecomposition of the covariance matrix of Y (i.e., Y^TY) can be written as PDP^{-1} where P is the matrix of eigenvectors and D is the diagonal matrix with corresponding eigenvalues as elements. Let P^* denote the matrix of eigenvectors sorted in decreasing order of eigenvalues. Therefore, the transformed data Y^* can be written as $Y^* = YP^*$. We can now achieve dimensionality reduction by computing the proportion of variation explained by each feature using the corresponding eigenvalues followed by heuristically choosing a threshold. PCA is a linear dimensionality approach and is not well suited for multi-likelihood settings.

Classical PCA has some shortcomings. It is not probabilistic as it has no likelihood model for the observed data. Moreover, computation of the covariance matrix and its associated eigendecomposition can be computationally intensive for large datasets with high dimensionality [Prasad and Bruce, 2008]. Another issue with classical PCA is that it cannot handle missing data properly and is not robust to outliers [Kambhatla and Leen, 1997]. Hence, it is not suitable in a generative model setting.

2.2.2 Probabilistic Principal Component Analysis

Probabilistic Principal Component Analysis (PPCA), proposed by Tipping and Bishop [1999], is a generalisation of the classical PCA that tries to overcome its shortcomings. It incorporates a probabilistic model and obtains a linear projection by maximising the likelihood. PPCA can be formulated as a latent variable model as in Figure 2.2. Assume a D-dimensional dataset

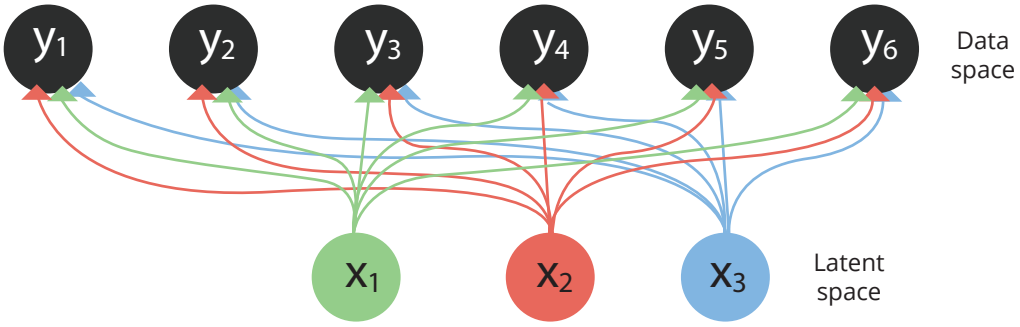


Figure 2.2: Visualisation of a latent variable model

$\boldsymbol{Y}$ of N points, i.e., $\boldsymbol{Y} = \{\boldsymbol{y}_n\}_{n=1}^N$ such that $\boldsymbol{y}_n \in \mathbb{R}^D$ and is centred. We can denote each latent variable corresponding to each data point as $\boldsymbol{x}_n \in \mathbb{R}^Q$ such that $Q \leq D$. Also, let $\boldsymbol{W} \in \mathbb{R}^{D \times Q}$ denote the principal axes or weights. We can write the likelihood for an individual data point as

$$\begin{aligned} p(\boldsymbol{y}_n|\boldsymbol{W}, \sigma^2) &= \int p(\boldsymbol{y}_n|\boldsymbol{x}_n, \boldsymbol{W}, \sigma^2) p(\boldsymbol{x}_n) d\boldsymbol{x}_n \\ p(\boldsymbol{x}_n) &= N(\boldsymbol{x}_n|0, \boldsymbol{I}) \\ p(\boldsymbol{y}_n|\boldsymbol{x}_n, \boldsymbol{W}, \sigma^2) &= N(\boldsymbol{y}_n|\boldsymbol{W}\boldsymbol{x}_n, \sigma^2 \boldsymbol{I}_D) \end{aligned}$$

where σ^2 is the noise and we assume an isotropic Gaussian noise model. To solve for $\boldsymbol{W}$ we assume that $\boldsymbol{y}_n$ is i.i.d and maximise the likelihood for all data points

$$p(\boldsymbol{Y}|\boldsymbol{W}, \sigma^2) = \prod_{n=1}^{N} p(\boldsymbol{y}_n|\boldsymbol{W}\boldsymbol{x}_n, \sigma^2 \boldsymbol{I}_D).$$

Marginalising out the latent variables, the distribution for each point can be written as

$$\boldsymbol{y}_n \sim N(0, \boldsymbol{W}\boldsymbol{W}^T + \sigma^2 \boldsymbol{I}_D).$$

The parameters are optimised to obtain the maximum likelihood. We can say that the classical PCA is a limiting case of PPCA when the covariance becomes infinitesimally small, i.e., $\sigma^2 \to 0$.

2.2.3 Gaussian process latent variable model

Lawrence [2004] proposed the Gaussian process latent variable model (GPLVM) as a generalisation of PPCA. It is an unsupervised learning algorithm that

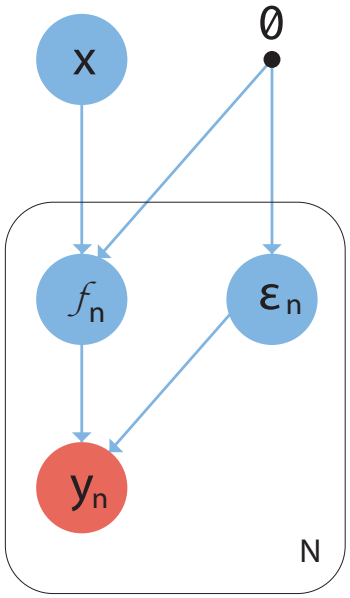


Figure 2.3: Graphical model of GPLVM. The arrows represent the dependency relations. The red circle corresponds to observed variables while the blue circle corresponds to latent variables.

extends PPCA by making use of a less restrictive covariance function that allows for non-linear mappings. Building upon the derivation of PPCA and following Lawrence [2004], we can obtain the GPLVM formulation.

A prior distribution is defined for $\boldsymbol{W}$ as $p(\boldsymbol{W}) = \prod_{i=1}^{D} N(\boldsymbol{w}_i|0, \alpha^{-1}\boldsymbol{I})$. From the PPCA derivation, instead of integrating out the latent variables $\boldsymbol{X}$, the principal axes $\boldsymbol{W}$ is marginalised to give the marginal likelihood for $\boldsymbol{Y}$,

$$p(\boldsymbol{Y}|\boldsymbol{X}, \sigma) = \frac{1}{(2\pi)^{\frac{DN}{2}}|\boldsymbol{K}|^{\frac{D}{2}}} \exp\left(-\frac{1}{2}\operatorname{tr}(\boldsymbol{K}^{-1}\boldsymbol{Y}\boldsymbol{Y}^T)\right),$$

where tr corresponds to the trace of a matrix and $\boldsymbol{K} = \alpha\boldsymbol{X}\boldsymbol{X}^T + \sigma^2\boldsymbol{I}$. Hence, the marginal likelihood that is being optimised can be interpreted as the product of D independent Gaussian processes where $\boldsymbol{K}$ is given by the linear covariance function. Therefore to obtain the GPLVM formulation, a non-linear covariance function is introduced which corresponds to a non-linear mapping from latent space to data space. A common choice for the process prior is the Radial Basis Function kernel. The marginal likelihood is

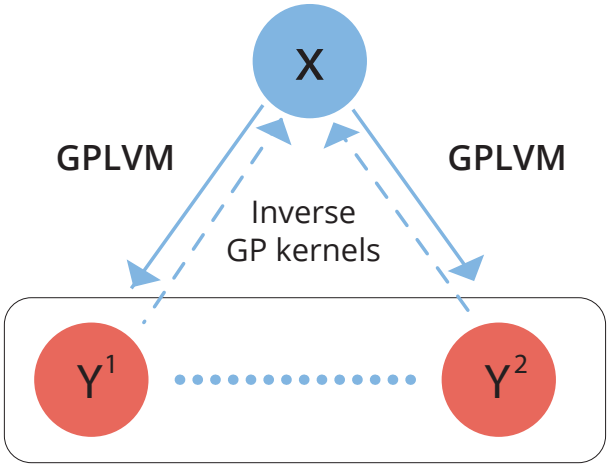


Figure 2.4: Graphical model of a shared GPLVM

jointly marginalised with respect to the latent variables, $\boldsymbol{X}$ and other parameters. In this study, we make use of the automatic relevance determination Radial Basis Function (ARD RBF) kernel that allows for separate length scales in each dimension of the latent space.

Hence, the optimisation problem can be written as,

$$\{\hat{\boldsymbol{X}}, \hat{\boldsymbol{\theta}}\} = \arg\max_{\boldsymbol{X},\boldsymbol{\theta}} p(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{\theta})$$

where $\boldsymbol{\theta}$ corresponds to the kernel parameters and $\hat{\boldsymbol{X}}$ as well as $\hat{\boldsymbol{\theta}}$ corresponds to the optimal values of the latent variables and kernel parameters respectively.

2.2.4 Shared GPLVM

Clinical patient records comprise of data from disparate sources which can be explained by different likelihoods. Therefore, we learn a shared GPLVM [Shon et al., 2006; Ek et al., 2007] that learns a shared latent representation over the different data likelihoods.

Extending the single likelihood specification of Li and Chen [2016], we can consider a case where N records comprise of data from two different likelihoods: $\boldsymbol{Y}^1 \in \mathbb{R}^{N\times D}$ and $\boldsymbol{Y}^2 \in \mathbb{R}^{N\times L}$ such that the total dimensionality for each record is $D + L$. Shared GPLVM tries to learn a common low

dimensional representation $\boldsymbol{X}$ for $\boldsymbol{Y}^1$ and $\boldsymbol{Y}^2$. In other words, $\boldsymbol{Y}^1$ and $\boldsymbol{Y}^2$ are generated by a GP from a shared latent representation $\boldsymbol{X}$. Figure 2.4 gives a graphical representation of the described scheme. The likelihood of the shared GPLVM can be written as,

$$p(\boldsymbol{Y}^1, \boldsymbol{Y}^2 | \boldsymbol{f}, \boldsymbol{X}, \boldsymbol{\theta}) = \prod_{i=1}^{N} p(\boldsymbol{y}_i^1 | \boldsymbol{f}_1, \boldsymbol{x}_i, \boldsymbol{\theta}) p(\boldsymbol{y}_i^2 | \boldsymbol{f}_2, \boldsymbol{x}_i, \boldsymbol{\theta})$$

where $\boldsymbol{f} = (\boldsymbol{f}_1, \boldsymbol{f}_2)$ and $\boldsymbol{\theta}$ are the hyperparameters of the GP.

Shared GPLVM which was proposed by Shon et al. [2006], assumed that all the data-spaces had a Gaussian likelihood. In this thesis, we extend this to support a composite/heterogeneous likelihood setting. In other words, we propose a framework that learns a shared latent representation $\boldsymbol{X}$ for $\boldsymbol{Y}^1$ and $\boldsymbol{Y}^2$ which may be from different and/or non-Gaussian likelihoods. Figure 2.3 illustrates the model as a plate diagram.

2.3 Variational inference

Approximate inference algorithms are used in models where inference (such as, marginal inference) is intractable. Sampling-based inference algorithms are quite computationally intensive and hence, may not always be feasible. Variational inference is a popular, alternate approach that recasts inference as an optimisation problem. This allows for the use of efficient, off-the-shelf optimisers and GPU acceleration. In other words, given a class of tractable probability distributions $\mathcal{Q}$, variational inference tries to solve an optimisation problem that yields a $q \in \mathcal{Q}$ that is most similar to the distribution p. The probability distribution p is intractable.

According to the original literature on variational inference [Jordan et al., 1999], we need to choose a family of approximating distributions ($\mathcal{Q}$) and an objective function that captures the similarity between q and p. The Kullback-Leibler (KL) divergence [Kullback and Leibler, 1951] is used as it provides a measure of difference between the two distributions. Let $\boldsymbol{x} = \{x_1, x_2, ..., x_m\}$ be a set of latent variables and $\boldsymbol{y} = \{y_1, y_2, ..., y_n\}$ be the set of observation points. Following Blei et al. [2017], we can consider each $q(\boldsymbol{x}) \in \mathcal{Q}$ as a candidate approximation to $p(\boldsymbol{x})$. Since we are trying to find the best candidate that minimises the KL divergence to the true distribution,

the inference is now recast to finding the optimal $q^*(\boldsymbol{x})$,

$$q^*(\boldsymbol{x}) = \arg\min_{q(\boldsymbol{x})\in\mathcal{Q}} \mathrm{KL}[q(\boldsymbol{x})||p(\boldsymbol{x}|\boldsymbol{y})] \tag{2.4}$$

$$\begin{aligned}\mathrm{KL}[q(\boldsymbol{x})||p(\boldsymbol{x}|\boldsymbol{y})] &= \mathbb{E}_{q(\boldsymbol{x})}[\log q(\boldsymbol{x})] - \mathbb{E}_{q(\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{y})] \\ &= \mathbb{E}_{q(\boldsymbol{x})}[\log q(\boldsymbol{x})] - \mathbb{E}_{q(\boldsymbol{x})}[\log p(\boldsymbol{x},\boldsymbol{y})] + \log p(\boldsymbol{y}) \end{aligned} \tag{2.5}$$

Hence, the objective cannot be computed as $\log p(\boldsymbol{y})$ is hard to compute. We optimise an alternative objective function called the evidence lower bound (ELBO), that is equivalent to the negative KL divergence obtained above up to a constant ($\log p(\boldsymbol{y})$ which is a constant with respect to $q(\boldsymbol{x})$).

$$\begin{aligned}\mathrm{ELBO} &= \mathbb{E}_{q(\boldsymbol{x})}[\log p(\boldsymbol{x},\boldsymbol{y})] - \mathbb{E}_{q(\boldsymbol{x})}[\log q(\boldsymbol{x})] \qquad (2.6)\\ &= \mathbb{E}_{q(\boldsymbol{x})}[\log p(\boldsymbol{x})] + \mathbb{E}_{q(\boldsymbol{x})}[\log p(\boldsymbol{y}|\boldsymbol{x})] - \mathbb{E}_{q(\boldsymbol{x})}[\log q(\boldsymbol{x})] \\ &= \mathbb{E}_{q(\boldsymbol{x})}[\log p(\boldsymbol{y}|\boldsymbol{x})] - \mathrm{KL}[q(\boldsymbol{x})||p(\boldsymbol{x})]. \qquad (2.7)\end{aligned}$$

Therefore, it can be observed from Equation (2.7) that maximising the ELBO corresponds to minimising the KL divergence in Equation (2.5). Also, from Equation (2.7) it can be observed that the first term is an expected likelihood and the second term encourages densities close to the prior. Moreover, using the derivation of the ELBO (i.e, substituting Equation (2.6) in Equation (2.5) and rearranging), we can rewrite the original derivation as

$$\log p(\boldsymbol{y}) = \mathrm{KL}[q(\boldsymbol{x})||p(\boldsymbol{x}|\boldsymbol{y})] + \mathrm{ELBO}\,.$$

From Kullback and Leibler [1951], we know that $\mathrm{KL}[.] \geq 0$. Hence, $\log p(\boldsymbol{y}) \geq$ ELBO. Therefore, ELBO acts as a lower bound for the log evidence. This bound may be a good approximation of the marginal likelihood and hence may be used as a model selection criteria or a qualitative measure of the model fit [Ueda and Ghahramani, 2002; Bernardo et al., 2003], though there is no justification in theory [Blei et al., 2017].

The choice of the approximating family of distributions, $\mathcal{Q}$, has an effect on the quality of the approximation and optimisation complexity. A common choice is the mean-field variational family that is based on the mean-field theory [Xing et al., 2002]. It assumes that the latent variables are mutually independent and are described by distinct factors.

$$q(\boldsymbol{x}) = \prod_{j=1}^{m} q_j(x_j).$$

Hence, we can see that with this approximation each latent variable, x_j, is governed by its own distribution that has its own factors/variational parameters. These variational distributions can take any form depending on the random variable (a common choice being Gaussian) and its parameters will be optimised with respect to the ELBO. There are many alternatives for the approximating posterior distributions. Some richer alternatives that have been proposed involve a mixture model [Jaakkola and Jordan, 1998; Gershman et al., 2012] or a normalising flow [Rezende and Mohamed, 2015].

2.4 Sparse Gaussian processes

Sparse Gaussian processes (Sparse GPs) were proposed to overcome the intractability of GPs on large datasets. The time complexity of GPs scales as $O(n^3)$ which poses a significant problem. Some initial methods to overcome this was to use a subset of the dataset to approximate the kernel matrix as proposed in Williams and Seeger [2001]. The idea behind sparse GPs is to formulate an approximation using a set of inducing or support variables that are smaller than the original dataset to reduce the overall time complexity. For example, assume that we have m inducing variables. Sparse GPs try to reduce the time complexity from $O(n^3)$ which is intractable for large datasets to $O(nm^2)$ which is tractable. There are many strategies that have been proposed to select these variables/points [Quiñonero-Candela and Rasmussen, 2005; Herbrich et al., 2003; Smola et al., 2000; Williams and Seeger, 2001]. Consider, a dataset given by $\boldsymbol{Y} = \{\boldsymbol{y}_i\}_{i=1}^N$ with $\boldsymbol{y}_i \in \boldsymbol{\mathcal{Y}}$. The above strategies proposed to select a small subset of the dataset, $\boldsymbol{Z} \subset \boldsymbol{Y}$, and perform the GP computations using that subset. Snelson and Ghahramani [2006] proposed to generalise the set of inducing variables so that they do not have to be a subset of the dataset. Therefore, the inducing variables are chosen such that $\boldsymbol{Z} \in \boldsymbol{\mathcal{Y}}$ but not necessarily a subset of $\boldsymbol{Y}$. The location of these "auxiliary pseudo-inputs" can be inferred with kernel hyperparameters using continuous optimisation [Quiñonero-Candela and Rasmussen, 2005].

In this thesis, we make use of the approach proposed by Titsias [2009] which makes use of a variational method to jointly select inducing inputs and hyperparameters by maximising a lower bound to the log-marginal likelihood (ELBO). The inducing inputs are variational parameters which are chosen by minimising the KL divergence between the posterior GP approximation and the true GP approximation. From Titsias [2009], we can write

the approximation to the posterior process as

$$
\begin{aligned}
q(\boldsymbol{f}) &= \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}) \\
\boldsymbol{\mu} &= k(\boldsymbol{X}, \boldsymbol{Z}) k(\boldsymbol{Z}, \boldsymbol{Z})^{-1} \boldsymbol{m} \\
\boldsymbol{\sigma} &= k(\boldsymbol{X}, \boldsymbol{X}) - k(\boldsymbol{X}, \boldsymbol{Z})[k(\boldsymbol{Z}, \boldsymbol{Z})^{-1} - k(\boldsymbol{Z}, \boldsymbol{Z})^{-1} \boldsymbol{\Sigma} k(\boldsymbol{Z}, \boldsymbol{Z})^{-1}] k(\boldsymbol{Z}, \boldsymbol{X}),
\end{aligned}
$$

where $\boldsymbol{X}$ is the latent representation and $\boldsymbol{m}, \boldsymbol{\Sigma}$ and $\boldsymbol{Z}$ are variational parameters that are optimised to maximise the ELBO. The introduction of Z as a parameter of the variational approximation was a significant contribution by Titsias [2009] (i.e., a variational lower bound is maximised to get the set of inducing inputs).

2.5 Bayesian GPLVM

Bayesian GPLVM [Titsias and Lawrence, 2010], is a variational inference framework for GPLVMs. The core difference between this and the GPLVM described in Section 2.2.3, is that instead of optimising the latent variables, they are variationally integrated out and a lower bound on the exact marginal likelihood is optimised. It also makes use of the concepts of Sparse GPs discussed in Section 2.4. Hence, using the ideas discussed in Section 2.3 the model parameters are learnt through the maximisation of the variational lower bound.

However, the expected likelihood (the first term in Equation (2.7)) involves an analytically intractable integral. This is because in Equation (2.7) for the GPLVM, $\log p(\boldsymbol{y}|\boldsymbol{x})$ contains $\boldsymbol{x}$ in a highly non-linear manner inside the inverse of the covariance matrix [Titsias and Lawrence, 2010]. Recall that from Section 2.2.3, the covariance matrix is given by $\boldsymbol{K} = \alpha \boldsymbol{x}\boldsymbol{x}^T + \sigma^2 \boldsymbol{I}$. In Titsias and Lawrence [2010], the authors derive a closed-form lower bound for the expected likelihood for GPLVMs with a Gaussian likelihood. From Equation (2.7), the lower bound for the expected likelihood can be computed across D dimensions of the data and can be written as

$$
\widetilde{\mathcal{F}}(q) = \sum_{d=1}^{D} \widetilde{\mathcal{F}}_d(q).
$$

From the derivation in Titsias and Lawrence [2010] and further elaborated in Damianou et al. [2016], we get (assuming a Gaussian likelihood):

$$
\widetilde{\mathcal{F}}_d(q) \geq \log \left[\frac{\beta^{\frac{N}{2}} |K_{MM}|^{\frac{1}{2}}}{(2\pi)^{\frac{N}{2}} |\beta \Psi_2 + K_{MM}|^{\frac{1}{2}}} \exp^{-\frac{1}{2} y_d^T W y_d} \right] - \frac{\beta \Psi_0}{2} + \frac{\beta}{2} \operatorname{tr}(K_{MM}^{-1} \Psi_2), \tag{2.8}
$$

where β is the inverse variance parameter (i.e. $\beta = \sigma^{-2}$) and $W = \beta I_N - \beta^2 \Psi_1 (\beta \Psi_2 + K_{MM})^{-1} \Psi_1^T$. The $\mathbf{\Psi}$ statistics, $\mathbf{\Psi} = (\Psi_0, \Psi_1, \Psi_2)$ are given by

$$\begin{aligned}\Psi_0 &= \mathrm{tr}(\mathbb{E}_{q(\boldsymbol{x})})[K_{NN}] \\ \Psi_1 &= \mathbb{E}_{q(\boldsymbol{x})}[K_{NM}] \\ \Psi_2 &= \mathbb{E}_{q(\boldsymbol{x})}[K_{MN} K_{NM}],\end{aligned}$$

where K_{NN}, K_{NM} and K_{MM} are the $N \times N$, $N \times M$ and $M \times M$ covariance matrices defined by the kernel function respectively. The $\mathbf{\Psi}$ statistics can be obtained analytically for the ARD RBF kernel. Summing Equation (2.8) over the dimension D and substituting in Equation (2.7) for the ELBO, we obtain a closed-form variational lower bound of the GPLVM with a Gaussian likelihood.

However, such a closed-form lower bound cannot be derived in our multi-likelihood setting. In this thesis, we make use of a sampling-based variational inference using numerical quadrature.

2.6 Numerical integration

Numerical integration refers to a family of algorithms that is used to compute the value of a definite integral. The main objective of numerical integration is to perform an approximate integration where the definite integral is approximated by a linear combination of the values of the integrand [Davis and Rabinowitz, 2007]. This can be formulated as,

$$\int_b^a f(x)dx \approx w_1 f(x_1) + w_2 f(x_2) + ... + w_m f(x_m)$$
$$-\infty \leq a \leq b \leq +\infty$$

where $\{x_1, x_2, ..., x_m\}$ are the abscissas that lie in the integration interval and $\{w_1, w_2, ..., w_m\}$ are the weights. In this thesis, we make use of the Gauss-Hermite quadrature.

Gauss-Hermite quadrature is a popular choice for numerical integration [Liu and Pierce, 1994]. It is used to effectively approximate integrals of the form:

$$\int_{-\infty}^{\infty} f(x) \exp(-x^2) dx \approx \sum_{i=1}^{m} w_i f(x_i) \tag{2.9}$$

where the nodes x_i are zeros of the m^{th} order Hermite polynomial and w_i are the corresponding weights. The equation in (2.9) can be re-expressed as,

$$\int_{-\infty}^{\infty} f(x) \exp(-x^2) dx = \int_{-\infty}^{\infty} f(t) \phi(t; \mu, \sigma) \tag{2.10}$$

Therefore, the sampling nodes would be given as $t_i = \mu + \sqrt{2}\sigma x_i$ and the weights would be $w_i/\sqrt{\pi}$. In this thesis, we make use of the three-point Gauss-Hermite quadrature (i.e., we perform the evaluation of the integrand at three points).

2.7 Recognition models

GPLVMs are inherently smooth mappings from latent space to data space [Lawrence, 2004]. This smoothness constraint means that if two data points are separated by a small distance in latent space then, the distance between the corresponding points in the data space is also small. In other words, two points that are far-apart in data space will not be embedded close together in latent space. However, this does not provide a locality preserving constraint that ensures that two points that are close together in data space will end up being embedded close together in the latent space [Lawrence and Quiñonero-Candela, 2006]. It is important to strike a balance between ensuring that dissimilar data points are embedded far apart and similar data points are embedded close to each other. GPLVM inherently ensures the former, however additional constraints can be incorporated to provide for the latter as well. We make use of the idea of recognition models (or back-constraints) [Bui and Turner, 2015] to create a non-linear, fully probabilistic and locality preserving dimensionality reduction technique.

We apply an approach based on the neuroscale algorithm [Lowe and Tipping, 1997]. In this algorithm, the smooth mapping from data space to latent space is explicitly incorporated by constraining the latent points to be a function of the input points, that is, $\boldsymbol{x}_{nq} = \boldsymbol{g}_q(\boldsymbol{y}_n; \boldsymbol{w})$ where $\boldsymbol{x}_{nq}$ is the q^{th} dimension of the latent representation of the n^{th} data point $\boldsymbol{y}_n$, $\boldsymbol{g}(.)$ is a mapping function and $\boldsymbol{w}$ are its corresponding parameters. Any mapping function can be used as long as the derivative of the outputs with respect to the parameters can be computed [Lawrence and Quiñonero-Candela, 2006]. As proposed in Bui and Turner [2015], we use a recognition model for $\boldsymbol{g}(.)$.

Multi-layer perceptrons (MLPs) are a popular class of Artificial Neural Networks that comprise of at least three layers: input layer, hidden layer

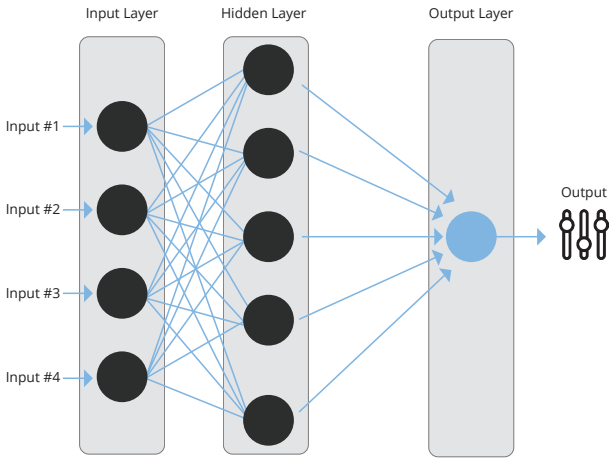


Figure 2.5: A simple feedforward multilayer perceptron

and output layer. Figure 2.5 depicts a simple fully connected feedforward MLP with a single hidden layer and 5 hidden units in the hidden layer. Each layer, except for the input layer, has an activation function which introduces a nonlinearity into the system (for example, *tanh*, *sigmoid*, *RELU*, etc;). At least one hidden layer is required to make MLPs a universal approximator [Goodfellow et al., 2016]. The term feedforward is used to emphasise the fact that the information flows only in one direction: from input layer to the output layer via the hidden layers. In a fully connected neural network, the output of each node in the hidden and output layers depends on the weighted sum of all the values from the previous layer. These weights are learnt through the backpropogation algorithm where the errors are fed back through the network and the weights are adjusted appropriately using an optimisation algorithm. In other words, the backpropogation algorithm can be considered as a special case of the chain-rule of derivation. We make use of the feedforward MLP as the recognition model. For a layer h with input values to the layer U, input weights W and bias b, we can write the general equation for the output of the layer as

$$\text{layer}(h; W, U, b) = \sigma_h \left(\underbrace{W^T}_{D_{out} \times D_{in}} \underbrace{U}_{D_{in} \times 1} + \underbrace{b}_{D_{out} \times 1} \right), \tag{2.11}$$

where σ_h is an activation for the layer h, and D_{in} as well as D_{out} correspond to the input and output dimensions of the layer respectively. This computation can be easily vectorised for N input elements allowing for efficient computations. Moreover, multiple layers can be incorporated by nesting Equation

(2.11) as is done in Equation (2.12) so that the output of a previous layer is taken as the input for the next layer. The input weights W and bias b must be learnt.

The constraint on the latent variables with a single hidden layer can now be written as

$$\boldsymbol{g}_q(\boldsymbol{y}_n) = \sigma_{out}\left(\underbrace{W_{2,q}^T}_{1\times H}\left[\sigma_{hidden}\left(\underbrace{(\underbrace{W_1^T}_{H\times D}\underbrace{\boldsymbol{y}_n}_{D\times 1})}_{H\times 1} + \underbrace{b_1}_{H\times 1}\right)\right] + \underbrace{b_{2,q}}_{1\times 1}\right), \tag{2.12}$$

where $\boldsymbol{y}_n$ is the n^{th} data point of the D dimensional dataset $\boldsymbol{y}$, H is the number of hidden units in the hidden layer, W_1 is the matrix of weights connecting the input layer to the hidden layer, b_1 captures the bias of the hidden layer, $W_{2,q}$ refers to the weights connecting the H hidden nodes to the q^{th} node of the output layer, $b_{2,q}$ refers to the q^{th} bias element of the output layer, σ_{hidden} is the element-wise activation function for the hidden layer and σ_{out} is the element-wise activation function for the output layer. The number of hidden units dictates the smoothness of the mapping.

By constraining the latent points to a smooth mapping from data space, small distances in data space will be small in latent space [Lawrence and Quiñonero-Candela, 2006]. Therefore, a constrained likelihood is optimised with constraints preserving the locality of data points. The use of a neural network allows for the introduction of efficient mini-batching through latent parameter sharing as well as predictions on unseen data.

2.8 Optimisation and automatic differentiation

To obtain the latent representation that best represents our data, we need to fit the model to the data by finding the parameters that give the highest ELBO value. In other words, we need to optimise the parameters of the model in such a way that we maximise the value of the ELBO. Gradient descent is a popular algorithm that is used to perform such an optimisation. There are many flavours of gradient descent. Figure 2.6 gives a partial overview of the variety of algorithms developed for this purpose. Most algorithms are variants of the basic idea of stochastic gradient descent. In this thesis we make use of two of such variants.

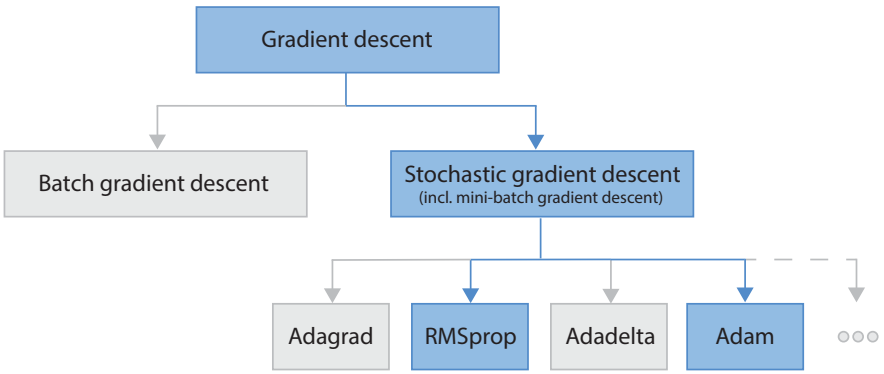


Figure 2.6: Gradient descent has many variants. We focus on Adam and RMSprop.

Gradient descent provides a way to minimise a cost function $J(\boldsymbol{\theta})$ (in our case we maximise the ELBO, which is equivalent to minimising the negative of the cost function $J(\boldsymbol{\theta})$) which has parameters $\boldsymbol{\theta}$ by updating the parameters in the opposite direction of the gradient of the objective function $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ with respect to the parameters [Ruder, 2016]. The size of the steps taken for each update is specified by the learning rate η. Vanilla gradient descent or batch gradient descent computes the gradient of the cost function with respect to the parameters over the entire training set for a single update which can be computationally intensive. This can be written as

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \cdot \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}).$$

Also, this would require the entire training set to be loaded into memory. Stochastic gradient descent (SGD) overcomes these shortcomings by performing a parameter update for each training example. This removes the redundancy in the gradient computation thereby making it faster and allows it to be an online algorithm. This can be written as

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \cdot \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}; y^i).$$

The problem with SGD is that the updates to the parameters have a high variance and hence the objective function may have rapid fluctuations. This may complicate the convergence as SGD may keep overshooting. To overcome the problem of overshooting the minimum, decaying learning rates can be used [Klein et al., 2009].

Mini-batch gradient descent tries to balance the benefits of both the approaches. By performing an update by computing the gradients over a mini-batch of l training examples, this approach reduces the variance of the parameter updates compared to SGD while still retaining computational efficiency by leveraging efficient matrix computations (vectorisation). Mini-batch gradient descent can be written as

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \cdot \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}; y^{i:i+l-1}),$$

where l is the size of the mini-batch. SGD can be considered as a specific case of mini-batch gradient descent where $l = 1$. SGD (and by extension mini-batch gradient descent) can be slow to converge or fluctuate rapidly depending on the choice of the learning rate. Hence, it is important to appropriately choose the learning rate. A common approach is to use learning rate schedules [Robbins and Monro, 1951] where the learning rates decay over time/epochs. The schedule still needs to be set before hand and it does not adapt to the characteristics of the data. The use of algorithms with adaptive learning rates solves these problems. Another issue with vanilla SGD is that the same learning rate is applied to all the parameters. This can have an adverse effect on convergence.

Momentum which was proposed in Qian [1999] tries to accelerate SGD by taking larger steps for dimensions whose gradients point in the same direction and smaller steps when the gradients change directions between consecutive time steps. They help SGDs to overcome the problem of getting stuck in areas where the surface curves more steeply in one direction that the other (ravines) [Sutton, 1986]

$$\begin{aligned} \boldsymbol{v_t} &= \gamma \boldsymbol{v_{t-1}} + \eta \cdot \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \\ \boldsymbol{\theta} &= \boldsymbol{\theta} - \boldsymbol{v_t}, \end{aligned}$$

where γ is the momentum term that is set to 0.9.

Adaptive learning rate algorithms adapt the learning rates for each parameter to better suit the data. Most adaptive learning rate algorithms keep a form of memory of the past gradients. This is then used to scale the learning rates accordingly. RMSprop is an adaptive learning rate algorithm that was proposed by Tieleman and Hinton [2012]. It involves a moving window of past squared gradients. Instead of inefficiently keeping track of the sum of a window of past squared gradients, it is approximated by a decaying average of all past squared gradients. RMSprop is immune to the problem of

radically diminishing learning rates. The running average of past squared gradients at time step, t can be written as

$$\mathbb{E}[\boldsymbol{g}^2]_t = \gamma\,\mathbb{E}[\boldsymbol{g}^2]_{t-1} + (1-\gamma)\boldsymbol{g}_t^2$$
$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \frac{\eta}{\sqrt{\mathbb{E}[\boldsymbol{g}^2]_t + \epsilon}}\boldsymbol{g}_t,$$

where $\boldsymbol{g}$ is a shorthand for $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$. Tieleman and Hinton [2012] recommends setting γ to 0.9, η to 0.001 and ϵ to 10^{-8}.

Adam (Adaptive Moment Estimation) is another adaptive learning rate algorithm that adapts the learning rate for each parameter by maintaining a form of history of past gradients [Kingma and Ba, 2014]. Adam maintains an exponentially decaying average of past gradients, in addition to maintaining an exponentially decaying average of past squared gradients like RMSprop. Following the notation of Kingma and Welling [2013], this can be written as

$$\boldsymbol{m}_t = \beta_1 \boldsymbol{m}_{t-1} + (1-\beta_1)\boldsymbol{g}_t$$
$$\boldsymbol{v}_t = \beta_2 \boldsymbol{v}_{t-1} + (1-\beta_2)\boldsymbol{g}_t^2,$$

where $\boldsymbol{m}_t$ and $\boldsymbol{v}_t$ are estimates for the first moment (decaying average of past gradients) and second moment (decaying average of past squared gradients) respectively that are initialised to zero and β_1 as well as β_2 are the decay rates. Kingma and Ba [2014] point out that $\boldsymbol{m}_t$ and $\boldsymbol{v}_t$ are biased to zero as they are initialised as zero vectors. They proposed a bias-corrected first and second moments

$$\hat{\boldsymbol{m}}_t = \frac{\boldsymbol{m}_t}{1-\beta_1^t}$$
$$\hat{\boldsymbol{v}}_t = \frac{\boldsymbol{v}_t}{1-\beta_2^t}.$$

Since the decay rates are raised to the power of t, $\hat{\boldsymbol{m}}_t \to \boldsymbol{m}_t$ for higher values of t. Hence, the bias correction affects the initial time steps the most. The parameters will be updated as

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \frac{\eta}{\sqrt{\hat{\boldsymbol{v}}_t} + \epsilon}\hat{\boldsymbol{m}}_t.$$

Following the recommendations of Kingma and Welling [2013], β_1, β_2 and ϵ are set to 0.9, 0.999 and 10^{-8} respectively.

There are many gradient descent optimisation algorithms that perform well in different problem settings. The choice of algorithm in most cases is

an empirical choice. In this thesis, we choose the RMSprop with momentum [Graves, 2013] for the fully Bayesian method and use Adam for the version with recognition models.

Automatic differentiation is a technique for computing exact derivatives to working precision (where the accuracy is limited by only floating point errors). In computer programs there are four main categories to compute the derivatives: manually compute the derivatives and code it up, numerical differentiation using finite differences, symbolic differentiation by translating the function into an expression, and automatic differentiation which allows the computation of gradients and Jacobians as well [Baydin et al., 2018]. In this thesis, we make use of automatic differentiation to compute gradients (and as a generalisation perform backpropogation) which is built into the Theano python library [Theano Development Team, 2016].

Automatic differentiation leverages the idea that complex functions are made up of primitive functions/operators. We can write any complicated function f as the composition of a sequence of primitive functions f_i (like a composition of simple building blocks), i.e., $f = f_1 \circ f_2 \circ ... f_i \circ ... \circ f_n$. Given that it would be quite simple to compute the derivative of the primitive functions, it would be easy to compute the derivative of the complex function (i.e., $\frac{df}{dx}$) through the chain rule. Hence, derivatives of functions of arbitrary order can be computed automatically. Depending on how the derivatives are decomposed using chain rule, automatic differentiation has two modes: forward mode and reverse mode.

In the forward mode, the derivatives of the parents ('complicated'/composite functions) are found in terms of the derivatives of their children ('primitive' functions). Consider an example of finding the partial derivative $\frac{\partial f}{\partial x}$ of a function, $f(x, y) = x \sin x + xy$. Figure 2.7 illustrates the corresponding computational graph that clearly depicts the nodes which are the arithmetic operations and the edges which depict the flow of information. For simplicity, we shall write $\frac{\partial f}{\partial x} = Df$. To solve the partial derivative, automatic

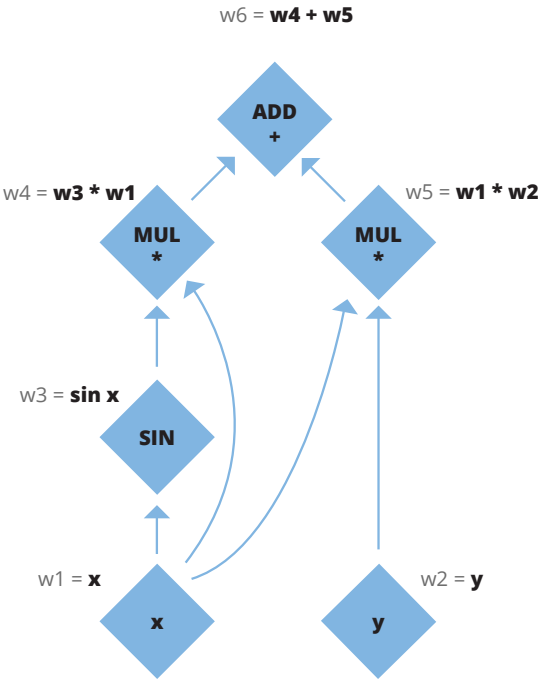


Figure 2.7: Example computational graph.

differentiation would unroll the function and apply chain rule as:

$$\begin{aligned}
Df &= D\boldsymbol{w}_6 \\
D\boldsymbol{w}_6 &= D(\boldsymbol{w}_4 + \boldsymbol{w}_5) = D\boldsymbol{w}_4 + D\boldsymbol{w}_5 \\
D\boldsymbol{w}_4 &= D(\boldsymbol{w}_3 \cdot \boldsymbol{w}_1) = \boldsymbol{w}_1 D\boldsymbol{w}_3 + \boldsymbol{w}_3 D\boldsymbol{w}_1 \\
D\boldsymbol{w}_3 &= D\sin \boldsymbol{w}_1 = \cos \boldsymbol{w}_1 \cdot D\boldsymbol{w}_1 \\
D\boldsymbol{w}_5 &= D(\boldsymbol{w}_1 \cdot \boldsymbol{w}_2) = \boldsymbol{w}_1 D\boldsymbol{w}_2 + \boldsymbol{w}_2 D\boldsymbol{w}_1 \\
D\boldsymbol{w}_2 &= D\boldsymbol{y} \\
D\boldsymbol{w}_1 &= D\boldsymbol{x} \\
D\boldsymbol{x} &= 1, \boldsymbol{y} = 0
\end{aligned}$$

Hence, it is clear that the value of $\frac{\partial f(x,y)}{\partial x}$ depends only on $x, y, D\boldsymbol{w}_1$ and $D\boldsymbol{w}_2$. This neatly demonstrates the elegance of automatic differentiation.

The backward mode is similar, but the derivatives of the child nodes are found in terms of their parent nodes instead. Hence, the information would flow in the direction opposite to that of the previous method. Backward mode is said to be more efficient than forward mode automatic differenti-

ation when we are finding the gradients with respect to a large number of parameters and hence, reverse mode is the primary technique in the form of the backpropogation algorithm [Baydin et al., 2018].

2.9 Gaussian mixture models and purity

Gaussian mixture models (GMM) is a density estimation algorithm that is represented as a weighted sum of Gaussian component densities [Reynolds, 2015]. In other words, it is a parametric probability density function that tries to model the data by identifying a mixture of multi-dimensional Gaussian probability distributions. Moreover given the probabilistic nature, GMMs can be used as a probabilistic clustering technique. This is done by assigning the data points to the Gaussian component that will maximise the posterior probability given the data. Following Reynolds [2015], the likelihood of a GMM with M components can be be written as,

$$p(\boldsymbol{x}|\theta) = \sum_{i=1}^{M} w_i \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$$

where $\boldsymbol{x}$ is the D dimensional latent vector in our case, w_i are the mixture weights, θ is the model parameters and $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ are the mean and covariance of the i^{th} component respectively. Moreover, $\sum_{i=1}^{M} w_i = 1$.

The choice of covariance matrix has an effect on the computational efficiency as well as the degrees of freedom in the shapes of the clusters. The covariance matrix could be chosen as full rank or constrained to be diagonal. Using diagonal covariance Gaussians is more computationally efficient but the orientation is constrained to align with the axis. The full rank covariance matrix allows for arbitrary orientation. The GMM parameters are estimated using the expectation-maximisation (EM) algorithm and this has been implemented in many standard libraries (see [Bishop, 2006]).

We make use of GMMs to discover clusters in the latent low-dimensional representation of the data. Moreover, we have no prior knowledge on the number of clusters that may be present. The optimal number of components M in a GMM that would best describe the latent representation can be estimated by computing the BIC score (Bayesian information criterion) [Schwarz et al., 1978] for GMMs with different number of components (a range of M) and choosing the one with the lowest BIC score. We extend the implementation provided in Pedregosa et al. [2011].

The purity score can be used to evaluate the quality of the resulting clusters in datasets for which the true labels are known. To compute the purity score, each cluster is first assigned the label that occurs the most. Then, the accuracy is computed by calculating the average number of data points that were currently assigned to a cluster [Amigó et al., 2009]

$$\text{purity} = \frac{1}{N}\sum_{k=1}^{K}\max_{j}|c_k \cap l_j|,$$

where N is the number of latent points, $C = \{c_1, c_2, ..., c_k, ..., c_K\}$ are the K clusters where each c_k corresponds to a cluster (or set of indices) and $L = \{l_1, l_2, ..., l_j, ..., l_J\}$ are the J categories or ground truth labels where each l_j corresponds to a set of indices. Higher purity signifies a better quality clustering.

2.10 Variational Bayesian Gaussian mixture models

This method is a variant of the Gaussian mixture model where we make use of variational inference and it can be seen as an extension of the expectation-maximisation method described in Section 2.9. We make use of this approach for the analysis of the clinical data. In this complementary approach, we maximise a lower bound on the model evidence and obtain an approximate posterior distribution over the parameters of the Gaussian mixture distribution. We do not need to specify the number of components in advance (i.e., the effective number of components can be inferred from the data). Moreover, it allows the incorporation of prior information leading to more stability and less dependence on the number of components [Nasios and Bors, 2006].

We follow the approach of Blei et al. [2006] by making using of the Dirichlet Process as a prior probability distribution on the clustering. For this approach we only need to specify the upper bound on the number of components. The optimal number of components can be obtained by observing the mixture weights as the method tends to set some weights closer to zero, if the pre-defined number of maximum components is large. We make use of the implementation provided in Pedregosa et al. [2011].

2.11 Summary of related work

Principle Component Analysis (PCA) is one of the most popular dimensionality reduction techniques [Jolliffe, 2011]. It is most commonly derived as a standardised linear projection which maximises the variance in the projected space [Hotelling, 1933]. The probabilistic reformulation of PCA, Probabilistic Principal Component Analysis, proposed by Tipping and Bishop [1999] incorporates a probabilistic model and arrives at a linear projection after the maximisation of the likelihood.

Gaussian process are powerful models that can perform tasks such as classification and regression and are popular algorithms in machine learning [Rasmussen and Williams, 2006]. Our method performs Bayesian nonlinear dimensionality reduction in a composite likelihood setting by building upon the GPLVM framework proposed by Lawrence [2004], which can be considered as a form of probabilistic reformulation of Principle Component Analysis [Lawrence, 2004]. Our work is closely related to [Gal et al., 2015] and [Wu et al., 2017]. However, the methods proposed in those papers focus on homogeneous data from a single likelihood distribution. To be precise, Gal et al. [2015] focuses on a categorical distribution and Wu et al. [2017] focuses on Poisson distributed data. Shon et al. [2006] proposed a generalisation of the GPLVM model that can handle multiple observation spaces (albeit with Gaussian likelihoods) where the observation spaces are linked by a lower dimensional latent variable space. This was extended by Ek et al. [2007] for 3D human pose estimation by incorporating constraints to the latent space. However, we try to maintain local clustering in the low dimensional latent space by incorporating back-constraints using the input data as in Lawrence and Quiñonero-Candela [2006] and Bui and Turner [2015].

Variational inference was introduced to GPLVMs in Titsias and Lawrence [2010]. Variational inference seeks to approximate the true posterior distribution by minimising the Kullback-Leibler divergence between the true posterior and a surrogate distribution. It can be seen as transforming a complex inference problem into a high-dimensional optimisation problem [Jordan et al., 1999; Wainwright et al., 2008]. Hoffman et al. [2013] improved the efficiency of variational inference by proposing an algorithm called stochastic variational inference that incorporated stochastic optimisation into variational inference. To overcome the intractability in our setting, we make use of a variant of stochastic variational inference, called sampling-based variational inference [Paisley et al., 2012; Rezende et al., 2014; Kingma and

Welling, 2013; Titsias and Lázaro-Gredilla, 2014]. We perform integration approximation using Gauss-Hermite quadrature.

More recently, Moreno-Muñoz et al. [2018] proposed an extension of the multi-output Gaussian processes that can handle heterogeneous outputs. Their approach is similar to ours but they focus on multi-output prediction (supervised learning) rather than identifying an appropriate latent representation (unsupervised learning).

2.12 Data description

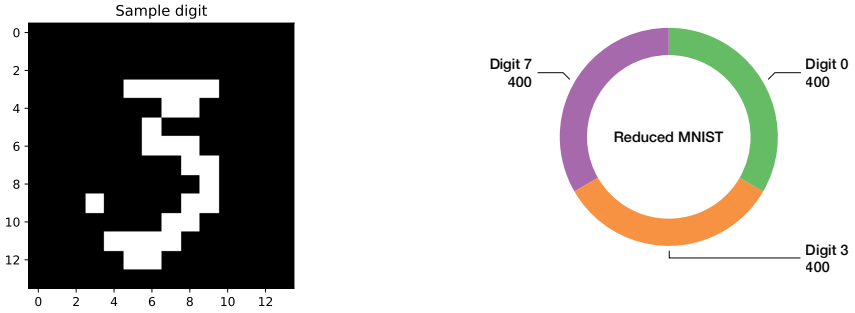
In this section, we describe the data that we will test our model on. The results of our analysis on these datasets can be found in Section 4.

2.12.1 Toy datasets

We first test our model on three toy datasets for which we have the real class labels. These datasets comprise of a single likelihood and are used to show that our model can provide good results in the simple settings. To demonstrate our model on real-world clinical heterogeneous data (multi-likelihood setting), we make use of the dataset described in Section 2.12.2. The toy datasets discussed in this section provide a simple means of validating our method.

2.12.1.1 Reduced MNIST dataset

We take a subset of the MNIST dataset [LeCun et al., 1998] by choosing 3 digits and randomly sampling 400 instances of each digit class. This gives us a total of 1200 digits. The original data is in grey scale and each pixel value ranges from 0 to 255. We binarise this data by applying a threshold such that values greater than 125 are taken as 1 and values less than or equal to 125 are taken as 0. We add some noise to the image by randomly corrupting 10% of the pixel values. The original MNIST dataset comprises of $28 \times 28 = 784$ pixels per digit. We perform down-sampling on this by performing 2×2 pooling of the binarised data. This reduces the data to $14 \times 14 = 196$ pixels per digit. Figure 2.8(a) visualises a digit from the binarised and down-sampled dataset that we will be used to evaluate our method. We will make use of the binomial likelihood. Hence, the dataset we will use has $N = 1200$ and $D = 196$.



(a) (b)

Figure 2.8: Dataset composition after preprocessing: (a) A sample digit after preprocessing and (b) Class composition of the reduced dataset.

2.12.1.2 *E. coli* dataset

We make use of the dataset provided by Horton and Nakai [1996]. It comprises of 7 attributes (excluding the sequence name) and the labels (or classes) are the localisation sites. Figure 2.9 visualises the composition of the dataset. In total, the original dataset comprises of 332 instances. We leave out the instances that belong to classes with less than 5 elements. In other words, we eliminate the elements belonging to *outer membrane lipoprotein*, *inner membrane lipoprotein* and *inner membrane - cleavable signal sequence* classes. Hence, we are left with $N = 327$, $D = 7$ and 5 localisation sites or classes. We will make use of the beta likelihood for this dataset.

2.12.1.3 Yeast dataset

To test the Gaussian likelihood, we make use of the Yeast dataset provided by [Horton and Nakai, 1996]. It comprises of 8 attributes (excluding the sequence name) and the labels (or classes) are the localisation sites. The class composition of the dataset is visualised in Figure 2.10. In the original dataset, there are a total of 1484 instances and 10 localisation sites (or class labels). We leave out the instances that belong to classes with less than 30 elements. Hence, we eliminate the instances that belong to *vacuolar*, *peroxisomal* and *endoplasmic reticulum lumen*. Hence, the dataset we test our method on has $N = 1429$, $D = 8$ and 7 localisation sites or classes.

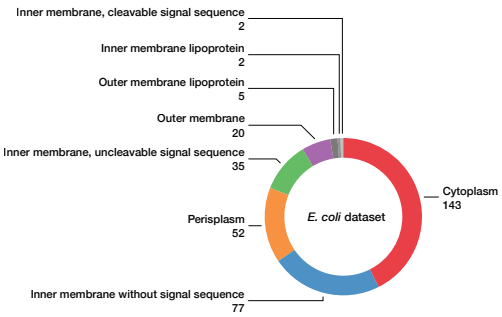


Figure 2.9: Class composition of the *E. coli* dataset.

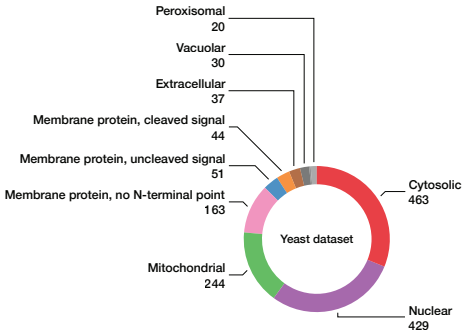


Figure 2.10: Class composition of the yeast dataset.

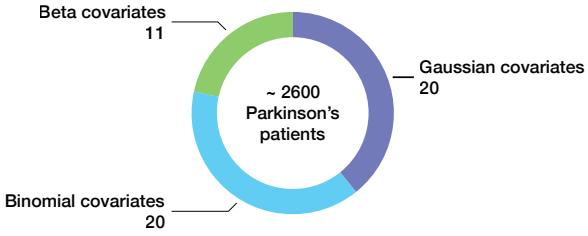


Figure 2.11: Class composition of the Parkinson's dataset.

2.12.2 Clinical data from Helsinki Biobank

We demonstrate our extended GPLVM model on heterogeneous clinical data. The data consisted of demographic information, diagnostic disease classifications and clinical laboratory tests of patients having Parkinson's disease treated in the HUS Helsinki University Hospital, Finland. The demographic information consisted of the patient's sex and age when the disease was first diagnosed. This was modelled with binomial and Gaussian likelihoods, respectively. The diagnostic information comprised of International Classification of Disease codes (ICD-10) at the categorical level (first three characters) obtained during a five-year follow-up period beginning at 6 months before the first Parkinson's diagnosis.

The disease codes were one-hot encoded into feature vectors and modelled with binomial likelihood. Laboratory measurements from blood (B), erythrocytes (E), plasma (P) or fasting plasma (fP), serum (S), urea (U) and leukocytes (L) were embedded into feature vectors as the median over the follow-up period. Notably, the laboratory data contained missing val-

ues. Variables expressing concentrations and percentages were modelled with Gaussian and beta likelihoods, respectively. Hence, our dataset comprised of approximately 2600 patients with 51-dimensional feature vectors consisting of 20 binomial, 20 Gaussian and 11 beta variables. 10% of the patients were held-out as test data. The class composition of this dataset is visualised in Figure 2.11.

Chapter 3

Methods

Our main contribution in this thesis is the extension of Gaussian process latent variable models (GPLVM) to produce low dimensional embeddings of heterogeneous datasets while preserving the similarities between the observations. In other words, we propose our method as a form of non-linear dimensionality reduction for heterogeneous data. The key principle in our model is to use the outputs of latent Gaussian processes to modulate the parameters of the different likelihoods through the use of link functions. We overcome the intractability introduced by the composite likelihoods by making use of sampling-based variational inference. Our model makes use of the inducing variable formalism for GPLVMs introduced in Titsias and Lawrence [2010] and computes a variational lower bound (ELBO) that is suitable for stochastic optimisation as in Gal et al. [2015]. Also, we make use of the idea of back-constraints to parameterise the variational inference and ensure that distant points in data space will be distant in latent space while preserving local similarities [Lawrence and Quiñonero-Candela, 2006]. The use of back-constraints through a recognition model allows our method to scale to larger datasets by allowing the use of mini-batching [Bui and Turner, 2015]. We demonstrate the applicability of our proposed method on clinical data of Parkinson's disease. Our method can be seen as a generalisation of Gal et al. [2015] for heterogeneous data while scalable to larger datasets. In this chapter, we discuss our method in detail.

3.1 Composite likelihood

Consider a generative model for a dataset $\boldsymbol{Y}$ with N observations (or patients in our case) and D variables of possibly different observation spaces (or as in our case, patient records from several disparate sources). The dataset can be

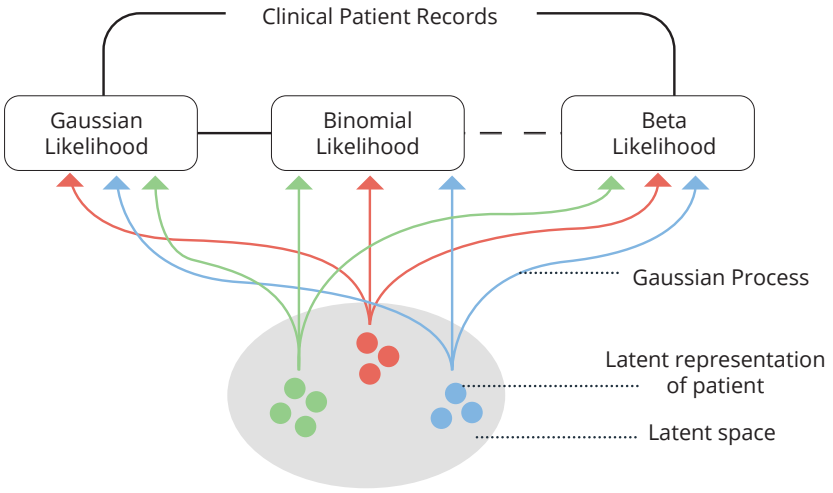


Figure 3.1: A simplified overview. Each point in the image corresponds to a patient, whose clinical observations comprise of data from different likelihoods, has been projected on to a two dimensional latent space. The colour coding in the latent space would correspond to disease sub-groups.

represented by a set of output functions $\mathcal{Y} = \{y_d(\boldsymbol{x}_n)\}_{d=1}^{D}$, where $\boldsymbol{x}_n \in \mathbb{R}^Q$ is the Q dimensional latent space for the n^{th} observation $\forall n = 1, ..., N$. Every observation in $\boldsymbol{Y}$ can be represented by a Q dimensional $\boldsymbol{x} \in \boldsymbol{X}$. The traditional GPLVM model considers the case where $y_d(\boldsymbol{x})$ is Gaussian distributed [Lawrence, 2004]. Moreover, Wu et al. [2017] and Gal et al. [2015] have proposed modifications to the GPLVM for Poisson and Categorical data respectively. In this paper, we propose a shared GPLVM [Ek et al., 2007] approach for which the observations in $\boldsymbol{Y}$ may be a mix of Gaussian, binary, beta, Poisson or categorical variables with several different distributions. Figure 3.1 gives a simplified illustration of our model. We shall assume that the distribution over $y_d(\boldsymbol{x}_n)$ is completely specified by a set of parameters $\boldsymbol{\vartheta}_d(\boldsymbol{x}_n) \in \psi^{P_d}$, where P_d is the number of parameters that define the d^{th} distribution and ψ is a generic domain for the parameters. We can think of each element $\vartheta_{d,p}(\boldsymbol{x_n})$ of parameter vector $\boldsymbol{\vartheta}_d(\boldsymbol{x}_n)$ as a non-linear transformation of a Gaussian process prior $\mathcal{F}_{d,p}$, such that $\vartheta_{d,p}(\boldsymbol{x}_n) = \phi_{d,p}(\mathcal{F}_{d,p}(\boldsymbol{x}_n))$ where $\phi_{d,p}(\cdot)$ acts as a link function (deterministic function) that maps the GP output to the appropriate domain for the parameter $\vartheta_{d,p}$.

To complete the generative model, we assign a Gaussian distribution prior with standard deviation σ_x^2 for the latent variables $x_{n,q} \in \boldsymbol{X}$. The model can be described by the following equations:

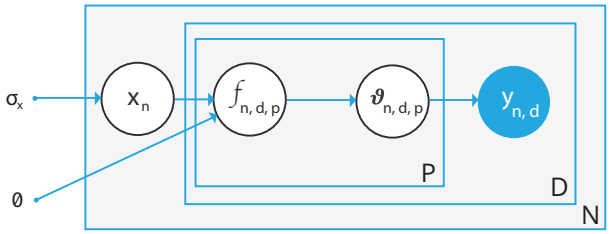


Figure 3.2: Plate diagram of the model. N, D and P corresponds to the number of observations, variables and parameters respectively. The shaded circle refers to an observed variable and un-shaded circles corresponds to the un-observed variables.

$$x_{n,q} \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_x^2) \tag{3.1}$$
$$\mathcal{F}_{d,p} \overset{\text{iid}}{\sim} \mathcal{GP}(0, k_d(\cdot)) \tag{3.2}$$
$$f_{n,d,p} = \mathcal{F}_{d,p}(\boldsymbol{x}_n) \tag{3.3}$$
$$\vartheta_{d,p}(\boldsymbol{x}_n) = \phi_{d,p}(f_{n,d,p}) \tag{3.4}$$
$$y_{n,d} \sim p(\cdot|\boldsymbol{\vartheta}_d(\boldsymbol{x}_n)), \tag{3.5}$$

where $n \in \{1, \ldots, N\}$, $q \in \{1, \ldots, Q\}$, $d \in \{1, \ldots, D\}$, $p \in \{1, \ldots, P_d\}$, $k_d(\cdot)$ is the GP kernel function, and $p(\cdot|\boldsymbol{\vartheta}_d(\boldsymbol{x}))$ denotes a generic likelihood function for the d^{th} variable. Our model uses the automatic relevance determination radial basis function kernel function.

To make the notation concrete, let us consider a case where each observation is comprised of two likelihoods and $D = 4$. Let the first two variables be Gaussian distributed and the last two correspond to count data which which we assume to follow Poisson distribution. In other words, $\mathcal{Y} = \{y_1(\boldsymbol{x}), y_2(\boldsymbol{x}), y_3(\boldsymbol{x}), y_4(\boldsymbol{x})\}$ where $y_1(\boldsymbol{x})$ and $y_2(\boldsymbol{x})$ are Gaussian distributed and $y_3(\boldsymbol{x})$ and $y_4(\boldsymbol{x})$ are Poisson distributed. We can say that $y_1(\boldsymbol{x})$ is modelled by two sets of parameters ($P_1 = 2$), $\boldsymbol{\vartheta}_1(\boldsymbol{x}) = [\vartheta_{1,1}(\boldsymbol{x}) \quad \vartheta_{1,2}(\boldsymbol{x})]$ corresponding to the mean and variance which are functions of $\boldsymbol{x}$ respectively. We can re-write this as $\boldsymbol{\vartheta}_1(\boldsymbol{x}) = [\phi_{1,1}(\mathcal{F}_{1,1}(\boldsymbol{x})) \quad \phi_{1,2}(\mathcal{F}_{1,2}(\boldsymbol{x}))]$ where $\phi_{1,1}(\cdot)$ would be the identity function and $\phi_{1,2}(\cdot)$ could be the exponential function to ensure that variance takes strictly positive values. Likewise, $y_2(\boldsymbol{x})$ would have a similar formulation. On the other hand, $y_3(\boldsymbol{x})$ and $y_4(\boldsymbol{x})$ would be modelled by the Poisson distribution which uses a single parameter ($P_3 = 1$) corresponding to the event rate (also written as λ). The outputs of $y_3(\boldsymbol{x})$ and $y_4(\boldsymbol{x})$ correspond to count variables that can take values

$y_3(\boldsymbol{x})$, $y_4(\boldsymbol{x}) \in \mathbb{N} \cup \{0\}$. Considering just $y_3(\boldsymbol{x})$ for now, we can say that it is modelled by $\boldsymbol{\vartheta}_3 = \vartheta_{3,1}(\boldsymbol{x}) = \phi_{3,1}(\mathcal{F}_{3,1}(\boldsymbol{x}))$. The rate parameter is restricted to positive real numbers, hence $\phi_{3,1}(\cdot)$ can be modelled by the exponential function that maps $\exp : \mathbb{R} \to (0, \infty)$. Likewise, $y_4(\boldsymbol{x})$ would have a similar formulation.

For our model, we assume that the outputs are conditionally independent given the vector of parameters given by $\boldsymbol{\vartheta}(\boldsymbol{x}) = [\boldsymbol{\vartheta}_1(\boldsymbol{x}), \boldsymbol{\vartheta}_2(\boldsymbol{x}), \boldsymbol{\vartheta}_3(\boldsymbol{x}), \ldots, \boldsymbol{\vartheta}_D(\boldsymbol{x})]$. Hence, the composite likelihood can be defined as

$$p(\boldsymbol{y}(\boldsymbol{x})|\boldsymbol{\vartheta}(\boldsymbol{x})) = p(\boldsymbol{y}(\boldsymbol{x})|\boldsymbol{f}(\boldsymbol{x})) = \prod_{d=1}^{D} p(y_d(\boldsymbol{x})|\boldsymbol{\vartheta}_d(\boldsymbol{x})), \tag{3.6}$$

where $\boldsymbol{f}$ contains realisations of all the GPs from Eq. 3.3. Previous works assume that all the variables are from the same observation space. In other words, a homogeneous dataset was represented by a single likelihood. We generalise the GPLVM model to $D \geq 1$ with possibly different likelihoods, thereby allowing it to create low-dimensional representations of heterogeneous datasets (or data from different observation spaces) that are represented by several different likelihoods while capturing the similarities between the observations.

3.2 Likelihood models

We consider the cases of Gaussian, binomial, beta, Poisson and categorical distributions in our analysis.

3.2.1 Gaussian distribution

For the Gaussian distribution, the distribution is specified by two parameters: mean and variance. The mean for each data point is obtained from the GPs, while the variance is a shared parameter that is optimised (and constrained to a positive value) to minimise the computational overhead. For the d^{th} measured variable this can be written as $\boldsymbol{\vartheta}_d = [\vartheta_{d,1}(\boldsymbol{x})]$ where $\vartheta_{d,1}(\boldsymbol{x})$ is the mean. Therefore, the mean is given by $\vartheta_{d,1}(\boldsymbol{x}) = \phi_{d,1}(\mathcal{F}_{d,1}(\boldsymbol{x}))$ where we choose $\phi_{d,1}(\cdot)$ to be the identity function.

3.2.2 Binomial distribution

A binomial distributions is specified by two parameters: number of trials and probability of success in each trial. In our case, for each data point the

number of trials is 1. Hence, this can be considered as a Bernoulli trial. We can write $\vartheta_{d,1}(\boldsymbol{x})$ as the probability of success for the d^{th} variable such that $\boldsymbol{\vartheta}_d = [\vartheta_{d,1}(\boldsymbol{x})]$. The probability of success would be given by $\vartheta_{d,1}(\boldsymbol{x}) = \phi_{d,1}(\mathcal{F}_{d,1}(\boldsymbol{x}))$ where we choose $\phi_{d,1}(\cdot)$ to be the *sigmoid* function (or *softmax* if considering success and failure separately).

3.2.3 Beta distribution

We re-parameterise the beta distribution in terms of mean, μ. Therefore, the two positive shape parameters (α and β) can be written as:

$$\begin{aligned} \alpha &= \nu\mu \\ \beta &= \nu(1-\mu), \end{aligned}$$

where ν is the inverse dispersion parameter which is a shared parameter that is optimised (and constrained to a positive value). Similar to the previous distributions, μ for each data point is given by $\vartheta_{d,1}(\boldsymbol{x}) = \phi_{d,1}(\mathcal{F}_{d,1}(\boldsymbol{x}))$ where we choose $\phi_{d,1}(\cdot)$ to be the CDF of the standard normal distribution (i.e., $\phi_{d,1}(\mathcal{F}_{d,1}(\boldsymbol{x})) = \Phi(\mathcal{F}_{d,1}(\boldsymbol{x}))$).

3.2.4 Poisson distribution

The Poisson distribution is specified by a single positive parameter known as the rate parameter (λ). Similar to the previous distributions, λ for each data point is given by $\vartheta_{d,1}(\boldsymbol{x}) = \phi_{d,1}(\mathcal{F}_{d,1}(\boldsymbol{x}))$ where we choose $\phi_{d,1}(\cdot)$ to be the exponential function.

3.2.5 Categorical distribution

For the categorical distribution, we make use of a formulation similar to Gal et al. [2015] which is a generalisation of the binomial distribution. In this case, the GPs produce the weights for each of the categories. We then make use of the *softmax* function to get probabilities for the categories in the range of $[0, 1]$.

Assume all categorical variables to have the same cardinality, K. Hence, $P_d = K$. For the d^{th} covariate of the n^{th} entry, we can write $\bar{f}_{n,d} = \{f_{n,d,1}, f_{n,d,2}, ..., f_{n,d,K}\}$. Following a similar notation to Gal et al. [2015],

we can write

$$y_{n,d} \sim \text{softmax}(\bar{f}_{n,d})$$
$$\text{softmax}(y_{n,d} = k; \bar{f}_{n,d}) = \text{categorical}\left(\frac{\exp(f_{n,d,k})}{\sum_{k'=1}^{K}\exp(f_{n,d,k'})}\right),$$

where categorical corresponds to the categorical distribution (or generalised Bernoulli distribution).

3.3 Auxiliary variables

The computational complexity of the Gaussian process models is reduced by the introduction of auxiliary variables or inducing inputs [Titsias, 2009]. We consider a set of M inducing inputs, $\boldsymbol{Z} \in \mathbb{R}^{M\times Q}$ that lie in the Q dimensional latent space. Their corresponding outputs in the input space would be $\boldsymbol{U} \in \mathbb{R}^{M\times D}$. According to Quiñonero-Candela and Rasmussen [2005], the auxiliary variables act as a support for the covariance function of the GP thereby allowing it to be evaluated on these points instead of the entire dataset. Hence, we can perform approximate inference in a time complexity of $\mathcal{O}(M^2N)$ instead of $\mathcal{O}(N^3)$ by evaluating the covariance function of the GP on the auxiliary variables instead of the entire dataset. Continuing the model description, we can write $u_{m,d} = \mathcal{F}_d(\boldsymbol{z}_m)$. Moreover, the joint distribution of $(\boldsymbol{f}_d, \boldsymbol{u}_d)$ is a multi-variate Gaussian distribution $N(0, \boldsymbol{K}_d([\boldsymbol{X}, \boldsymbol{Z}], [\boldsymbol{X}, \boldsymbol{Z}]))$. Further marginalising the inducing outputs leads to a joint distribution of the form $\boldsymbol{f}_d \sim \mathcal{N}(0, \boldsymbol{K}_d(\boldsymbol{X}, \boldsymbol{X})), \forall d$. Hence, the marginal likelihood of the data remains unchanged by the introduction of the auxiliary variables.

3.4 Variational inference

In our model, the marginal log-likelihood is intractable due to the presence of an arbitrary number of non-Gaussian likelihoods. Hence, we make use of variational inference to compute a lower bound of the log-likelihood (ELBO). Following Titsias and Lawrence [2010] and Gal et al. [2015], we obtain the ELBO (represented as $\mathcal{L}$) by applying Jensen's inequality with a variational distribution of the latent variables.

We consider a mean field approximation for the latent points $q(\boldsymbol{X})$ and

a joint Gaussian distribution for $q(\boldsymbol{U})$ as follows:

$$q(\boldsymbol{U}) = \prod_{d=1}^{D} \mathcal{N}(\boldsymbol{u}_d | \boldsymbol{\mu}_d, \boldsymbol{\Sigma}_d)$$
$$q(\boldsymbol{X}) = \prod_{n=1}^{N} \prod_{i=1}^{Q} \mathcal{N}(x_{n,i} | m_{n,i}, s^2_{n,i}).$$

To obtain the ELBO, we can first write the log-likelihood as,

$$\begin{aligned} \log p(\boldsymbol{Y}) &= \log \int p(\boldsymbol{X}) p(\boldsymbol{U}) p(\boldsymbol{F}|\boldsymbol{X}, \boldsymbol{U}) p(\boldsymbol{Y}|\boldsymbol{F}) d\boldsymbol{X} d\boldsymbol{F} d\boldsymbol{U} \\ &= \log \int \frac{q(\boldsymbol{X}, \boldsymbol{F}, \boldsymbol{U})}{q(\boldsymbol{X}, \boldsymbol{F}, \boldsymbol{U})} p(\boldsymbol{X}) p(\boldsymbol{U}) p(\boldsymbol{F}|\boldsymbol{X}, \boldsymbol{U}) p(\boldsymbol{Y}|\boldsymbol{F}) d\boldsymbol{X} d\boldsymbol{F} d\boldsymbol{U}. \end{aligned} \tag{3.7}$$

The variational approximation to the posterior distribution, $q(\boldsymbol{X}, \boldsymbol{F}, \boldsymbol{U})$, can be factorised as follows:

$$q(\boldsymbol{X}, \boldsymbol{F}, \boldsymbol{U}) = q(\boldsymbol{X}) q(\boldsymbol{U}) p(\boldsymbol{F}|\boldsymbol{X}, \boldsymbol{U}). \tag{3.8}$$

Substituting into Equation (3.7), we get:

$$\begin{aligned} \log p(\boldsymbol{Y}) = \log \int & q(\boldsymbol{X}) q(\boldsymbol{U}) p(\boldsymbol{F}|\boldsymbol{X}, \boldsymbol{U}) \\ & \cdot \frac{p(\boldsymbol{X}) p(\boldsymbol{U}) p(\boldsymbol{F}|\boldsymbol{X}, \boldsymbol{U}) p(\boldsymbol{Y}|\boldsymbol{F})}{q(\boldsymbol{X}) q(\boldsymbol{U}) p(\boldsymbol{F}|\boldsymbol{X}, \boldsymbol{U})} d\boldsymbol{X} d\boldsymbol{F} d\boldsymbol{U}. \end{aligned} \tag{3.9}$$

Jensen's inequality relates the value of a concave (or convex) function of an integral to the integral of the concave (or convex) function [Jensen et al., 1906]. Assume φ is a concave function and X is a random variable. By the Jensen's inequality for a concave function, we can write:

$$\varphi(\mathbb{E}[X]) \geq \mathbb{E}[\varphi(X)]. \tag{3.10}$$

In our model, we have $\varphi = \log$. Substituting this in Equation (3.10) and for a random variable X, we have:

$$\log(\mathbb{E}[X]) \geq \mathbb{E}[\log(X)]. \tag{3.11}$$

We can now apply the Jensen's inequality from Equation (3.11) to Equation (3.9):

$$\begin{aligned} \log p(\boldsymbol{Y}) \geq \int & q(\boldsymbol{X}) q(\boldsymbol{U}) p(\boldsymbol{F}|\boldsymbol{X}, \boldsymbol{U}) \\ & \cdot \log \frac{p(\boldsymbol{X}) p(\boldsymbol{U}) p(\boldsymbol{F}|\boldsymbol{X}, \boldsymbol{U}) p(\boldsymbol{Y}|\boldsymbol{F})}{q(\boldsymbol{X}) q(\boldsymbol{U}) p(\boldsymbol{F}|\boldsymbol{X}, \boldsymbol{U})} d\boldsymbol{X} d\boldsymbol{F} d\boldsymbol{U}. \end{aligned} \tag{3.12}$$

The Kullback-Leibler divergence [Kullback and Leibler, 1951] between $q(\boldsymbol{X})$ and $p(\boldsymbol{X})$ as well as between $q(\boldsymbol{U})$ and $p(\boldsymbol{U})$ can be written as,

$$\begin{aligned}\mathrm{KL}(q(\boldsymbol{X})||p(\boldsymbol{X})) &= \int q(\boldsymbol{X}) \log \frac{q(\boldsymbol{X})}{p(\boldsymbol{X})} d\boldsymbol{X} \\ \mathrm{KL}(q(\boldsymbol{U})||p(\boldsymbol{U})) &= \int q(\boldsymbol{U}) \log \frac{q(\boldsymbol{U})}{p(\boldsymbol{U})} d\boldsymbol{U}.\end{aligned}$$

Substituting the KL divergences in Equation (3.12) and unwrapping the remaining terms along the dimension d (i.e. the dimension of the data space) from their vectorised form, we get:

$$\begin{aligned}\log p(\boldsymbol{Y}) \geq & \overbrace{-\mathrm{KL}(q(\boldsymbol{X})||p(\boldsymbol{X})) - \mathrm{KL}(q(\boldsymbol{U})||p(\boldsymbol{U}))}^{(i)} \\ & + \underbrace{\sum_{d=1}^{D} \int q(\boldsymbol{X}) q(\boldsymbol{U}_d) p(\boldsymbol{f}_d|\boldsymbol{X}, \boldsymbol{U}_d) \cdot \log p(\boldsymbol{y}_d|\boldsymbol{f}_d) d\boldsymbol{X} d\boldsymbol{f}_d d\boldsymbol{U}_d}_{(ii)} = \mathcal{L}.\end{aligned} \tag{3.13}$$

This gives us a lower bound on $\log p(\boldsymbol{Y})$ similar to Equation (2.7) which was derived using the KL divergence instead. Comparing Equation (3.13) with Equation (2.7), we can see that (i) in Equation (3.13) corresponds to the KL divergence in Equation (2.7) (with the additional KL divergence for $\boldsymbol{U}$ to accommodate the inducing inputs) and (ii) in Equation (3.13) corresponds to the expectation of the log likelihood in Equation (2.7). We shall derive a lower bound on this as it is still intractable.

We marginalise $\boldsymbol{U}_d$ in the posterior distribution of $\boldsymbol{f}_d$ to obtain,

$$q(\boldsymbol{f}_d|\boldsymbol{X}) = \int p(\boldsymbol{f}_d|\boldsymbol{X}, \boldsymbol{U}_d) q(\boldsymbol{U}_d) d\boldsymbol{U}_d. \tag{3.14}$$

From Equation (3.14), we obtain a normal distribution with the following mean and variance,

$$\begin{aligned}q(\boldsymbol{f}_d|\boldsymbol{X}) = \mathcal{N}(&\boldsymbol{f}_d|\boldsymbol{K}_{NM}\boldsymbol{K}_{MM}^{-1}\boldsymbol{\mu}_d, \boldsymbol{K}_{NN} \\ &+ \boldsymbol{K}_{NM}\boldsymbol{K}_{MM}^{-1}(\boldsymbol{\Sigma}_d - \boldsymbol{K}_{MM})\boldsymbol{K}_{MM}^{-1}\boldsymbol{K}_{NM}^{T}),\end{aligned} \tag{3.15}$$

where $\boldsymbol{\mu}_d$ and $\boldsymbol{\Sigma}_d$ are the variational parameters and $\boldsymbol{K}_{NM}$ is the cross-covariance matrix computed over $\boldsymbol{Z}$ and $\boldsymbol{X}$. Similarly, $\boldsymbol{K}_{MM}$ as well as $\boldsymbol{K}_{NN}$ are the kernel matrices computed on $\boldsymbol{Z}$ and $\boldsymbol{X}$ respectively.

From Equation (3.15), we can write Equation (3.13) as:

$$\log p(\boldsymbol{Y}) \geq -\mathrm{KL}(q(\boldsymbol{X})||p(\boldsymbol{X})) - \mathrm{KL}(q(\boldsymbol{U})||p(\boldsymbol{U})) + \sum_{d=1}^{D} \int q(\boldsymbol{X}) q(\boldsymbol{f}_d|\boldsymbol{X}) \cdot \log p(\boldsymbol{y}_d|\boldsymbol{f}_d) d\boldsymbol{X} d\boldsymbol{f}_d. \tag{3.16}$$

To solve the integral over $\boldsymbol{X}$, we make use of Monte Carlo integration by drawing samples, $\boldsymbol{x}_i$ from $q(\boldsymbol{X})$. Hence from Equation (3.16), we can write the lower bound $\mathcal{L}$ as:

$$\mathcal{L} \approx -\mathrm{KL}(q(\boldsymbol{X})||p(\boldsymbol{X})) - \mathrm{KL}(q(\boldsymbol{U})||p(\boldsymbol{U})) + \frac{1}{N_x} \sum_{i=1}^{N_x} \sum_{d=1}^{D} \mathbb{E}_{q(\boldsymbol{f}_d|\boldsymbol{x}_i)}[\log p(\boldsymbol{y}_d|\boldsymbol{f}_d)], \tag{3.17}$$

where N_x corresponds to the number of samples drawn.

The variational expectation over the log likelihood, $\log p(\boldsymbol{y}_d|\boldsymbol{f}_d)$ is intractable. We solve this by making use of the Gauss-Hermite quadrature [Hensman et al., 2015]. Hence, we follow a sampling based approach [Paisley et al., 2012; Kingma and Welling, 2013; Titsias and Lázaro-Gredilla, 2014; Rezende et al., 2014] to compute the lower bound $\mathcal{L}$ as well as its derivatives with Gauss-Hermite quadrature.

Concretely, we transform the random variables to be sampled using the re-parameterisation trick introduced in Kingma and Welling [2013]. The transformation for $\boldsymbol{X}$ is as follows:

$$\boldsymbol{x}_n = \boldsymbol{m}_n + \boldsymbol{s}_n \boldsymbol{\epsilon}_n^{(x)}, \quad \boldsymbol{\epsilon}_n^{(x)} \sim \mathcal{N}(0, 1).$$

Also, $\boldsymbol{f}_d$ can be transformed as:

$$\boldsymbol{f}_{d,i} = \boldsymbol{a}_d + \boldsymbol{b}_d t_i,$$

where t_i are the zeros of the I^{th} order Hermite polynomial (we assume $I = 3$) as specified by the Gauss-Hermite quadrature and where $\boldsymbol{a}_d$ and $\boldsymbol{b}_d$ are specified from Equation (3.15):

$$\begin{aligned} \boldsymbol{a}_d &= \boldsymbol{K}_{NM} \boldsymbol{K}_{MM}^{-1} \boldsymbol{\mu}_d \\ \boldsymbol{b}_d \boldsymbol{b}_d^T &= \boldsymbol{K}_{NN} + \boldsymbol{K}_{NM} \boldsymbol{K}_{MM}^{-1} (\boldsymbol{\Sigma}_d - \boldsymbol{K}_{MM}) \boldsymbol{K}_{MM}^{-1} \boldsymbol{K}_{NM}^T. \end{aligned}$$

Hence, we can write the expectation for the log-likelihood as

$$\mathbb{E}_{q(\boldsymbol{f}_d|\boldsymbol{x}_n)}[\log p(\boldsymbol{y}_d|\boldsymbol{f}_d)] = \sum_{i=1}^{I} w_i q(\boldsymbol{f}_{d,i}|\boldsymbol{x}_n) \log p(\boldsymbol{y}_d|\boldsymbol{f}_{d,i}), \tag{3.18}$$

where in our case, we take $I = 3$ making Equation (3.18) a 3-point Gauss-Hermite quadrature and w_i are the suitably corresponding weights.

3.5 Variational recognition models

In the standard GPLVM model, there is no constraint that prevents two points which are close in data space to be embedded far apart in latent space [Lawrence and Quiñonero-Candela, 2006]. Moreover, the use of minibatch-based stochastic variational inference can be impractical for modest size datasets as achieving convergence can take a long time. This is because only the local parameters for a minibatch in each iteration are updated and the optimal $q(\boldsymbol{X})$ found for the other data points is ignored [Bui and Turner, 2015]. We borrow ideas from Bui and Turner [2015]; Lawrence and Quiñonero-Candela [2006]; Rezende et al. [2014] and parameterise the mean and covariance of the variational distribution over $q(\boldsymbol{X})$ using neural network based recognition models. Concretely, the mean and covariance of $q(\boldsymbol{x}_n)$ are obtained as the output of two feed-forward multi-layer perceptrons whose weights are trained by stochastic optimisation

$$q(\boldsymbol{x}_n|\boldsymbol{y}_n) = \mathcal{N}(\boldsymbol{x}_n|\mathcal{M}_{\omega_1}(\boldsymbol{y}_n), R_{\omega_2}(\boldsymbol{y}_n)^T R_{\omega_2}(\boldsymbol{y}_n)), \tag{3.19}$$

where $\mathcal{M}$ is the mean, R is the cholesky factor of the covariance and ω_1 and ω_2 are the network weights. Therefore, $\boldsymbol{m}_n = \mathcal{M}_{\omega_1}(\boldsymbol{y}_n)$ and $\boldsymbol{s}_n^2 = R_{\omega_2}(\boldsymbol{y}_n)^T R_{\omega_2}(\boldsymbol{y}_n)$. By parameterising the distribution over the latent variables by an inverse mapping from the observations, we are introducing a constraint that ensures that observations that are close in the data space will also be close in the latent representation. Moreover, the use of this formulation allows the efficient use of minibatching, thereby efficient stochastic optimisation. Specifically, the deep neural network weights, ω_1 and ω_2 act as global parameters that enable parameter sharing. Also, updating these parameters with respect to a data point in a minibatch also effects the latent representation of other data points. The gradients of these back-constraint parameters are obtained using the standard back-propagation algorithm.

The choice of weight initialisation for the deep neural networks can effect the training of the weights [Sutskever et al., 2013]. We make use of the weight initialisation described in Glorot and Bengio [2010] for both the networks.

3.6 Stochastic optimisation

The lower bound (ELBO) that needs to be optimised is valid across the data observations and hence, can be written as:

$$\mathcal{L} = -\sum_{n=1}^{N}\sum_{q=1}^{Q}\mathrm{KL}(q(x_{n,q})||p(x_{n,q})) - \sum_{d=1}^{D}\mathrm{KL}(q(\boldsymbol{u}_d)||p(\boldsymbol{u}_d)) + \sum_{n=1}^{N}\sum_{d=1}^{D}\left(\frac{1}{N_x}\sum_{i=1}^{N_x}\mathbb{E}_{q(f_{n,d,1}|\boldsymbol{x}_{n,i})\ldots q(f_{n,d,P_d}|\boldsymbol{x}_{n,i})}\Big[\log p(y_{n,d}|f_{n,d,1},\ldots,f_{n,d,P_d})\Big]\right), \tag{3.20}$$

where P_d is the number of parameters for the d^{th} likelihood.

We can make use of a suitable stochastic optimisation technique to minimise the ELBO. Concretely, the parameters we need to optimise for the fully Bayesian model (i.e, without the recognition model) include $\boldsymbol{m}_q$, $\boldsymbol{s}_q$, variational parameters $\boldsymbol{Z}$, $\boldsymbol{\mu}_d$, $\boldsymbol{\Sigma}_d$ and the hyper-parameters for the GP. For the optimisation, we make use of the RMSprop optimiser [Tieleman and Hinton, 2012].

To optimise the ELBO of the model with the variational recognition model (back constraints), the parameters we need to optimise include the recognition model weights (ω_1 and ω_2), variational parameters $\boldsymbol{Z}$, $\boldsymbol{\mu}_d$, $\boldsymbol{\Sigma}_d$ and the hyper-parameters for the GP. The optimisation is done using the Adam optimiser [Kingma and Ba, 2014].

Our method allows the computation of derivatives using automatic differentiation. We make use of Theano [Theano Development Team, 2016] for the inference implementation.

Chapter 4

Results

In this chapter, we first test our method on binomial, beta and Gaussian distributed data and analyse the effect of the choice of settings (that is, the number of inducing points and latent space dimensions). Then, we run our model on a clinical dataset from the Helsinki Biobank and visualise the results.

4.1 Toy datasets

We study the effect of the choice of number of inducing points and dimensions of the latent space on the ELBO and purity score. For each setting, we repeat the optimisation 5 times and choose the model with the best ELBO as the optimised model for that setting. The tile with a red box corresponds to the ELBO or purity score of the latent representation that is visualised. Each configuration is run for 2000 iterations.

4.1.1 Binomial distributed data

Figure 4.1 depicts the ELBO values obtained for different values of M (number of inducing points) and Q (dimensions of latent space) for the reduced MNIST dataset without using neural networks. From the figure, better ELBO values (darker region) are obtained in the middle region with latent dimensions between 8 to 16 and number of inducing points between 40 to 80. The evidence lower bound appears to degrade for higher values of latent space dimension and number of inducing points probably because of the increased complexity of the optimisation thereby requiring more iterations. Figure 4.2 measures the quality of the resulting clustering using the purity score described in Section 2.9 for different values of M and Q. The resulting

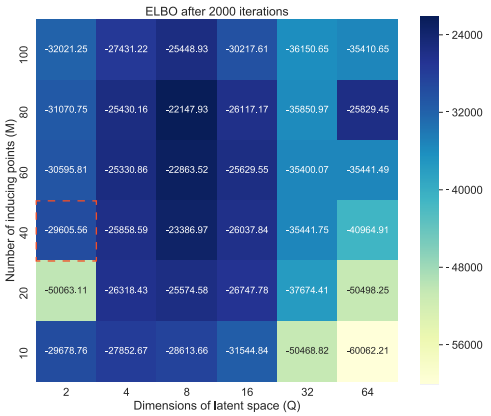


Figure 4.1: Heat map depicting the variation of ELBO without recognition models.

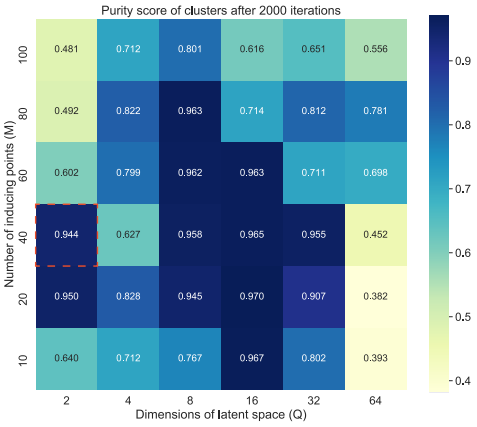


Figure 4.2: Heat map depicting the variation of purity score without recognition models.

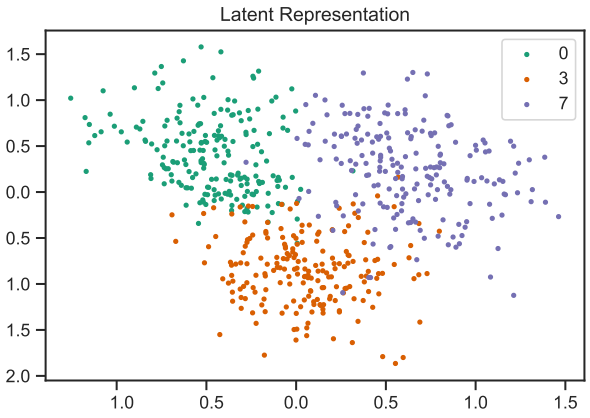


Figure 4.3: Latent representation with $Q = 2$ and $M = 40$ without recognition models.

heat map corroborates with the observations of 4.1 and shows a similar trend with higher purity scores for higher ELBO values. Figure 4.3 visualises the resulting latent representation for a 2 dimensional latent space. Each dot in Figure 4.3 corresponds to a single data point.

Similarly, we repeat our analysis using the same dataset but make use of the recognition models discussed in Section 3.5. Figure 4.4 visualises the variation of the ELBO while Figure 4.5 visualises the variation of the purity score with respect to the parameters. Also, in Figure 4.6 we visualise the

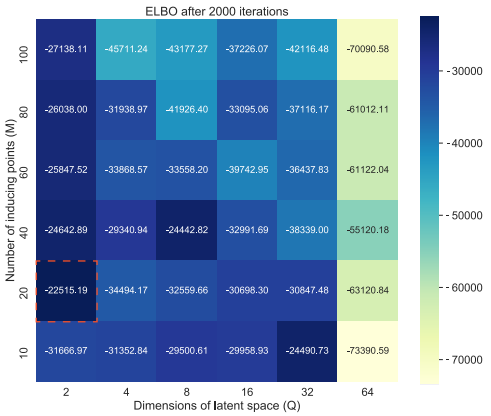


Figure 4.4: Heat map depicting the variation of ELBO.

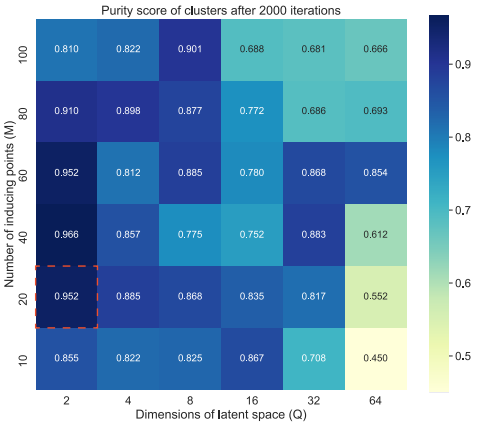


Figure 4.5: Heat map depicting the variation of purity score.

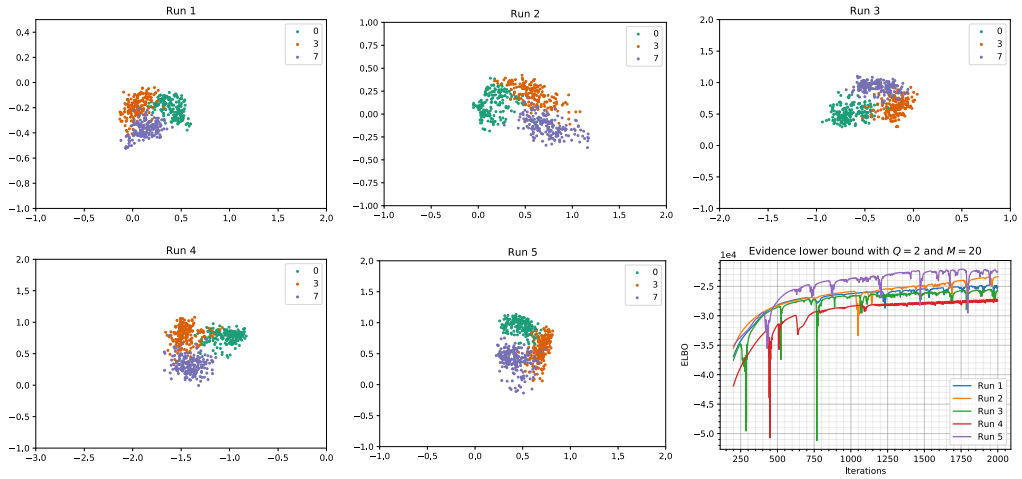


Figure 4.6: Latent representations and ELBO of the 5 optimisation runs with $Q = 2$ and $M = 20$ using recognition models on the binomial distributed data.

latent representation with respect to the parameters corresponding to the red box. Each dot corresponds to a data point and the colours correspond to the original class labels.

It can be observed that the ELBO and purity score follow similar trends with better results for smaller values of Q. This could be explained by the better expressive power of the recognition models. As discussed in Section

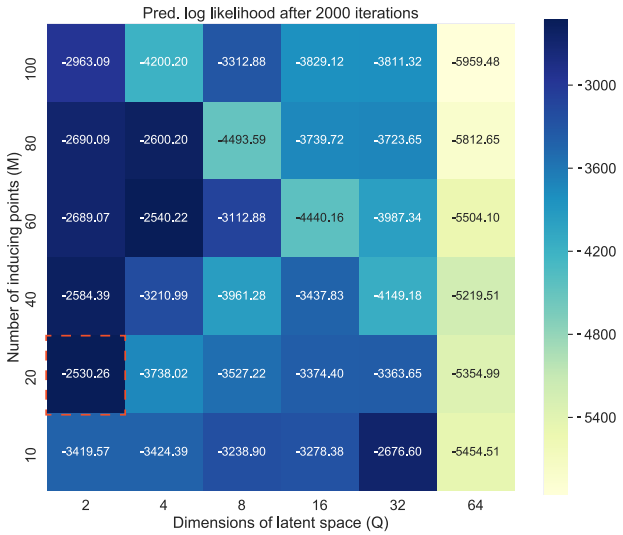


Figure 4.7: Heat map visualising the predictive log likelihood on test data

3.5, we can see from Figure 4.6 that the method tries to balance between similarity and dissimilarity preservation.

We also visualise the variation in the predictive log likelihood computed on some test data which is obtained by holding out approximately 10% of the training data. This can be seen in Figure 4.7. It is possible to make predictions on some test data easily with recognition models as the learnt neural network weights can be used to easily estimate the mean and variance for the variational distribution in time complexity $\mathcal{O}(N)$. In the model without the neural networks, we would have to optimise the entire model again to get the latent representation of test points and to obtain the predictive likelihood.

4.1.2 Beta distributed data

We performed a similar analysis using a dataset with beta distributed data. Figure 4.8 visualises the variations in the ELBO. As before, we perform 5 optimisation runs for each setting. We can see that higher lower bounds are achieved when using more inducing points. Moreover, using 2 or 4 latent dimensions give the best ELBO values. A similar trend can be observed in the purity scores. This is visualised in Figure 4.9.

Furthermore, we visualise the latent representation of the highlighted parameters (shown by the red box) and can observe that a good degree of clustering is achieved. The points in the latent space are coloured based on

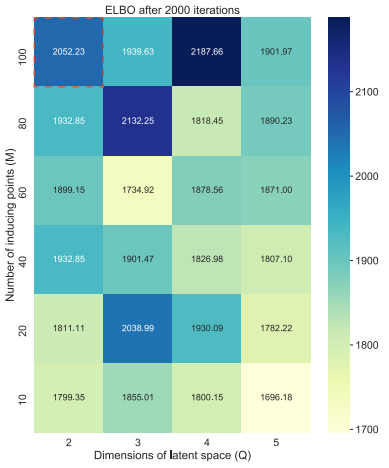


Figure 4.8: Heat map depicting the variation of ELBO without recognition models.

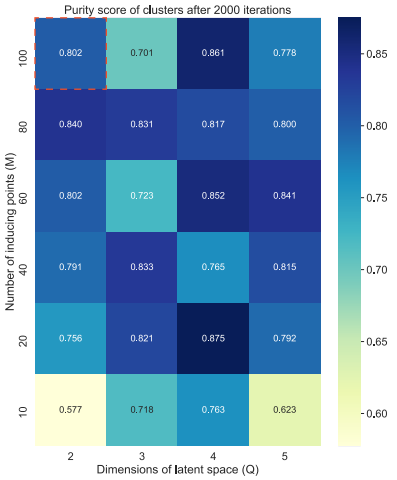


Figure 4.9: Heat map depicting the variation of purity score without recognition models.

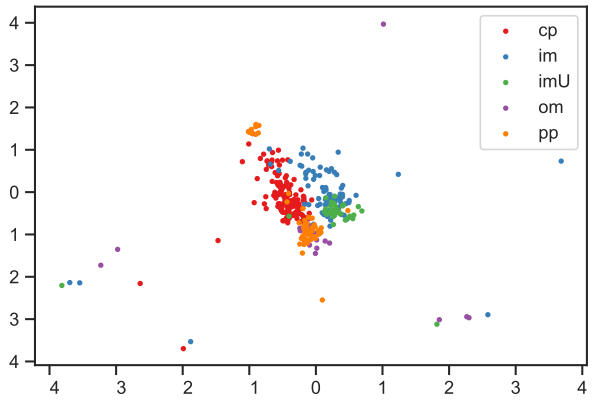


Figure 4.10: Latent representation with $Q = 2$ and $M = 100$ without recognition models.

the original class labels.

Similarly, we repeat the analysis using the recognition models as well on the same data. Figures 4.11 and 4.12 visualise the resulting variations in the ELBO and purity scores respectively. From these plots, we can see that higher purity and ELBO values are achieved with lower values of Q and with higher values of M. The resulting ELBO and purity score values for higher

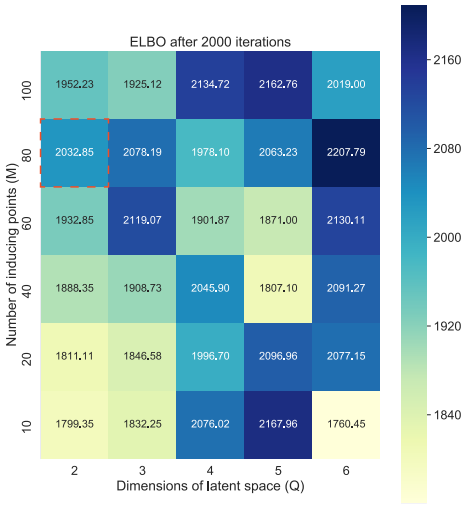


Figure 4.11: Heat map depicting the variation of ELBO.

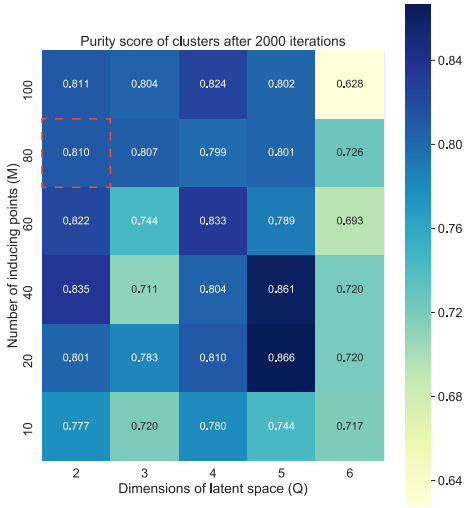


Figure 4.12: Heat map depicting the variation of purity score.

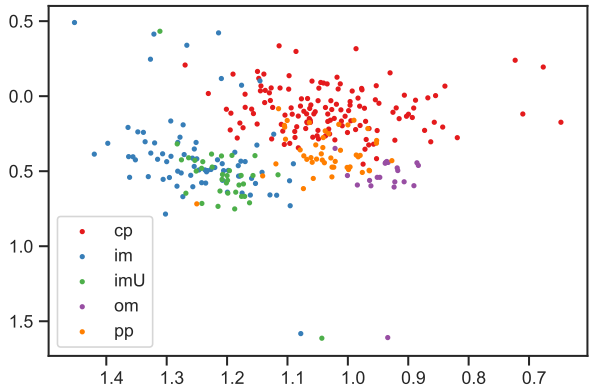


Figure 4.13: Latent representation with $Q = 2$ and $M = 80$.

values of Q are lower possibly because more iterations may be required.

The resulting latent space shown in Figure 4.13 shows effective clustering and achieves better separation of the classes while keeping data points that belong to the same class nearby in the latent space in comparison with Figure 4.10.

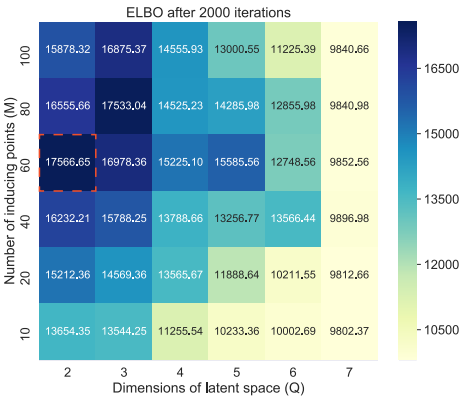


Figure 4.14: Heat map depicting the variation of ELBO without recognition models.

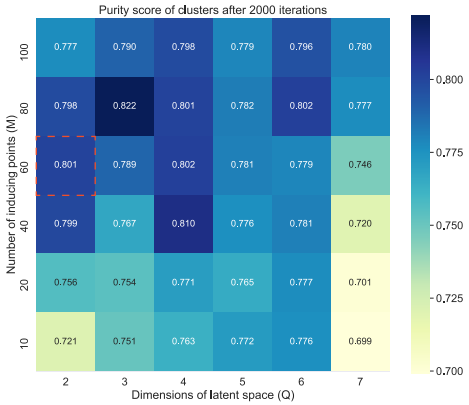


Figure 4.15: Heat map depicting the variation of purity score without recognition models.

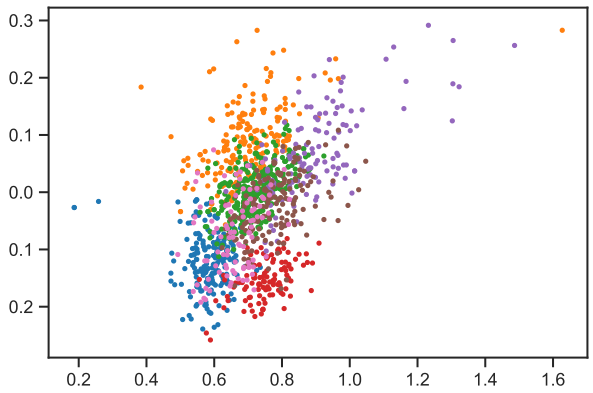


Figure 4.16: Latent representation with $Q = 2$ and $M = 60$ without recognition models.

4.1.3 Gaussian distributed data

We also studied the performance of our model on a dataset with Gaussian distributed data and visualised the resulting variations in the ELBO and purity score in Figures 4.14 and 4.15 respectively. A similar trend in the purity scores and ELBO can be observed. The latent representation of the dataset using the selected parameters can be seen in Figure 4.16. The points are coloured based on the labels from the original dataset.

We also analyse the performance with recognition models and visualise the resulting ELBO and purity score in Figures 4.17 and 4.18 respectively.

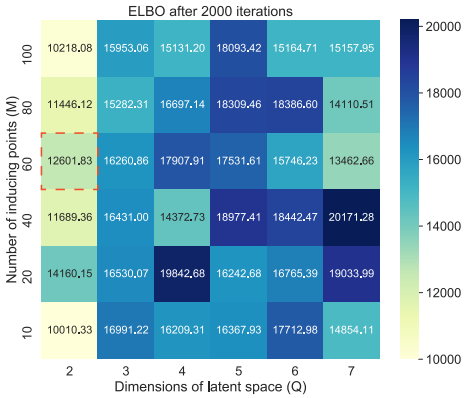


Figure 4.17: Heat map depicting the variation of ELBO.

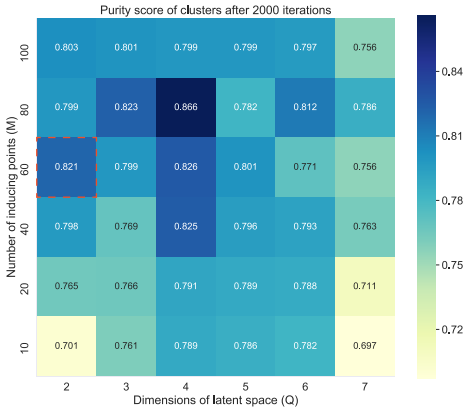


Figure 4.18: Heat map depicting the variation of purity score.

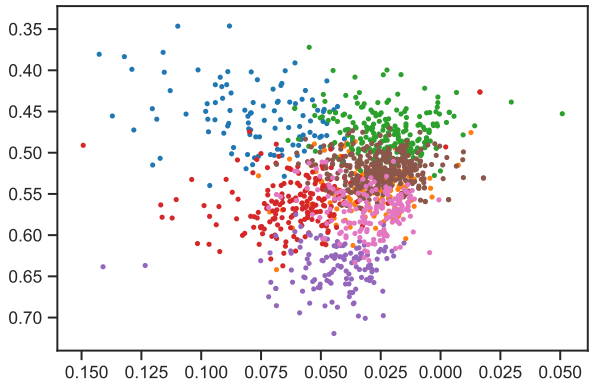
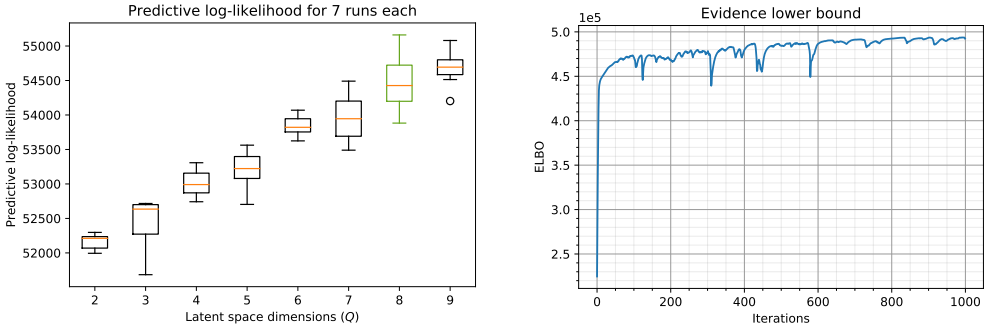


Figure 4.19: Latent representation with $Q = 2$ and $M = 60$.

From Figure 4.19, we can observe the latent representation of the parameters highlighted by the red box. By comparing Figure 4.15 and Figure 4.18, we can see that recognition models achieve better purity scores. The resulting latent space appears to achieve better clustering than Figure 4.16. The points in the latent space are coloured using the original class labels.

4.2 Composite likelihood clinical dataset

We first assessed the optimal latent dimensionality of the dataset. We did this by comparing the predictive log likelihood obtained using the test data with different latent dimensions (i.e., different values of Q). The algorithm



(a) (b)

Figure 4.20: Assessment of optimal latent dimensionality: (a) Predictive log-likelihood and (b) Evidence lower bound (ELBO).

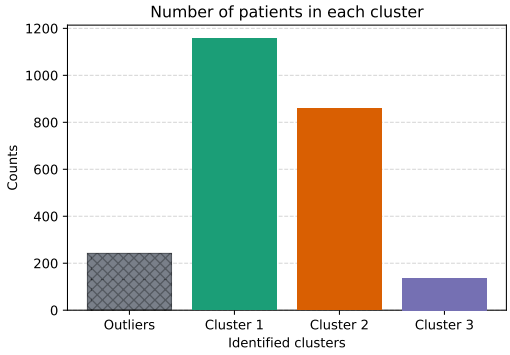


Figure 4.21: Analysis of the resulting clusters.

was run seven times per dimensionality and the prediction was done using the model having the largest ELBO over 1000 iterations. From Figure 4.20(a), we can see that the highest value is obtained for $Q = 8$. Also, the Figure 4.20(b) visualises the optimisation of the corresponding evidence lower bound (ELBO).

We then performed clustering on the latent embeddings obtained using $Q = 8$ with the Bayesian Gaussian mixture model as discussed in Section 2.10 to obtain the optimal number of latent clusters as well as the cluster membership of each sample. The result with the highest lower bound out of 20 initialisations was selected and the maximum number of clusters (i.e., the upper bound on the number of components) was set to 10. The final number of clusters was chosen by the algorithm and we excluded the clusters contain-

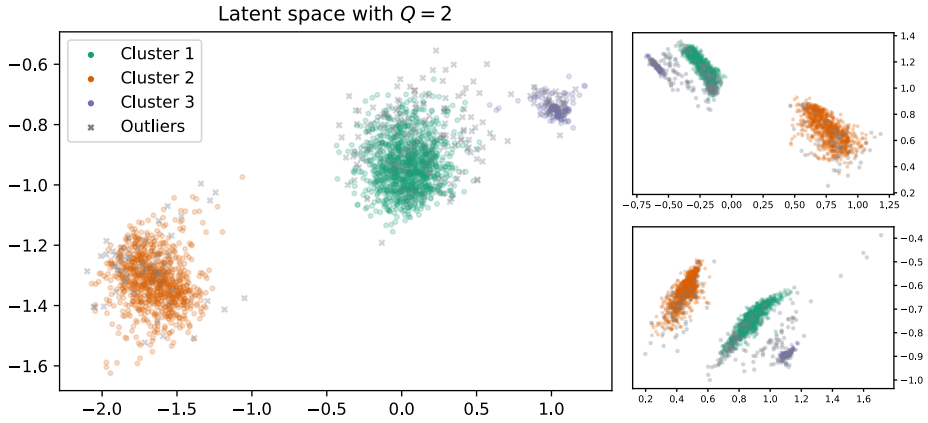


Figure 4.22: Latent representations of the patients from different runs

ing less than 5% of the patients (these were marked as outliers). Hence, a total of 3 latent clusters were identified and a histogram of the cluster membership can be seen in Figure 4.21. We used our method to obtain a latent embedding of the patients with $Q = 2$ for visualisation purposes and made use of the cluster membership obtained previously. Figure 4.2 visualises the latent representation of the patients from different optimisation runs.

Finally, we evaluated the differences in the characteristics of the clusters. We computed the logarithmic odds-ratio separately for each binomial variable between the values of samples belonging to a specific cluster and the rest of the data. For other variables, the t-Statistic was applied respectively. Thus, negative values indicate less frequent occurrence or smaller values than the rest of the data, and vice versa. For example, from Figure 4.23, cluster 3 characteristically includes younger people (observing the *age* covariate) who also have a lower frequency of other diagnoses than the rest of the patients in other clusters. Also from Figures 4.23 and 4.24, in clusters 1 and 2, age is not a determining factor but the clusters often show opposite characteristics both in prevalence of diagnoses and in laboratory measurements.

These results demonstrate the feasibility of our approach in finding patient subsets in a data-driven manner.

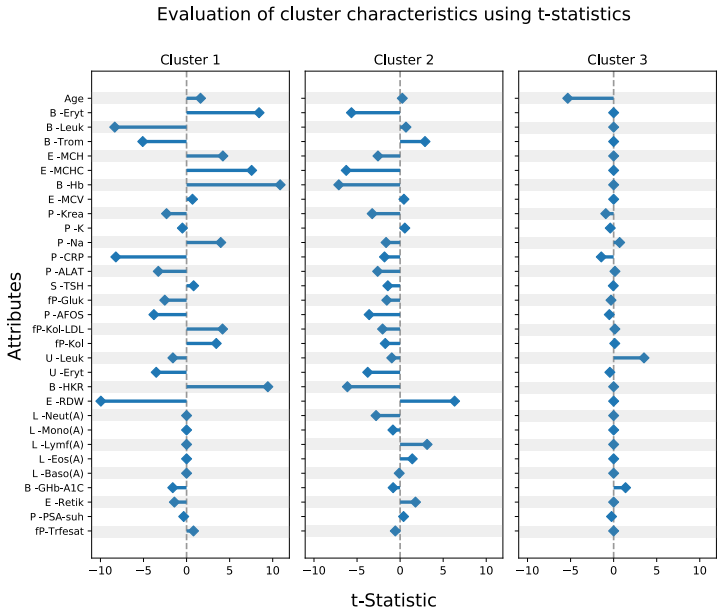


Figure 4.23: t-Statistics

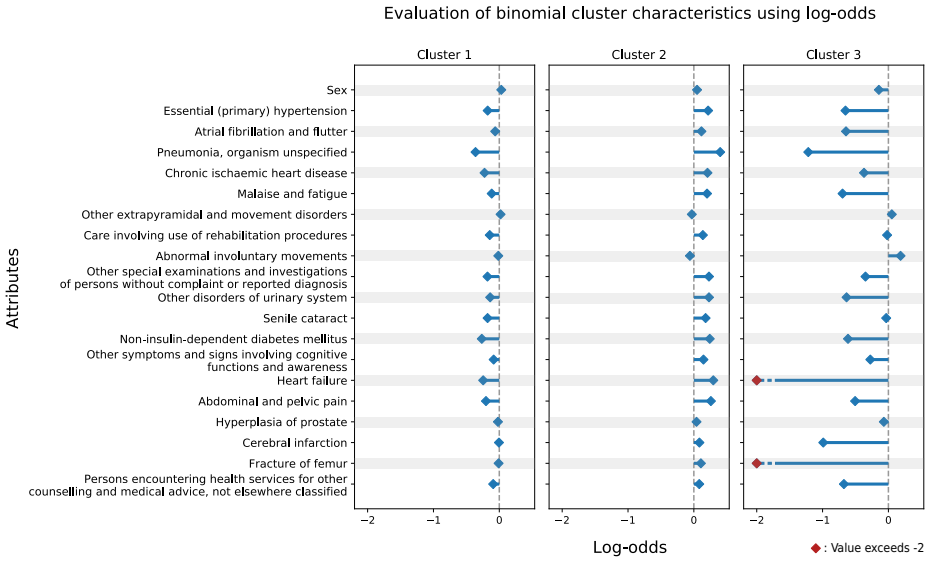
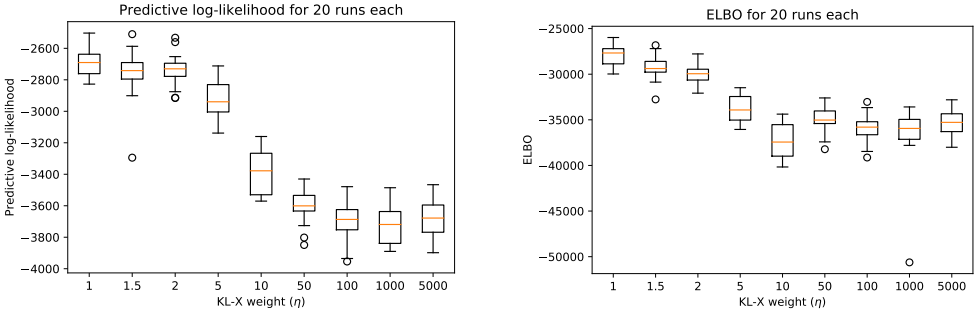


Figure 4.24: Log-odds ratios



(a) (b)

Figure 4.25: Assessment of optimal value for η: (a) Predictive log-likelihood and (b) Evidence lower bound (ELBO).

4.3 Origin-centred latent representations

It is possible that the resulting latent embeddings may not be centred about the origin even after the model seems sufficiently optimised. This can be observed in some of the visualisations of the latent embeddings using the toy datasets and clinical dataset. We propose a modification to our method to achieve origin-centred latent embeddings by leveraging the idea of introducing a hyper-parameter that balances the latent channel capacity and independence constraints with reconstruction accuracy as described in Higgins et al. [2017]. In this thesis, we shall call this hyper-parameter as η.

In this section, we shall demonstrate that using $\eta > 1$ results in embeddings that are centred about the origin but with qualitatively consistent results (i.e., qualitatively consistent with $\eta = 1$). From Equation (3.17), we can write the lower bound $\mathcal{L}$ with our suggested modification as follows:

$$\mathcal{L} \approx -\eta \overbrace{\mathrm{KL}(q(\boldsymbol{X})||p(\boldsymbol{X}))}^{\mathrm{KL}_{\boldsymbol{X}}} - \overbrace{\mathrm{KL}(q(\boldsymbol{U})||p(\boldsymbol{U}))}^{\mathrm{KL}_{\boldsymbol{U}}} + \frac{1}{N_x}\sum_{i=1}^{N_x}\sum_{d=1}^{D}\mathbb{E}_{q(\boldsymbol{f}_d|\boldsymbol{x}_i)}[\log p(\boldsymbol{y}_d|\boldsymbol{f}_d)], \tag{4.1}$$

where η is the new hyper-parameter. If $\eta = 1$, we get the same equation as Equation (3.17).

To obtain a centred embedding, the hyper-parameter η must be tuned. In Figure 4.25, we obtain the ELBO and predictive log-likelihood for various

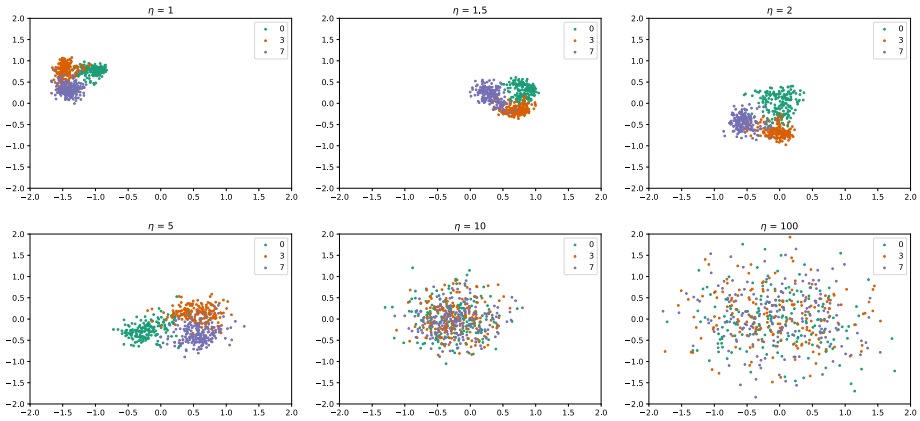


Figure 4.26: Latent representations of the digits using different values for the hyper-parameter η.

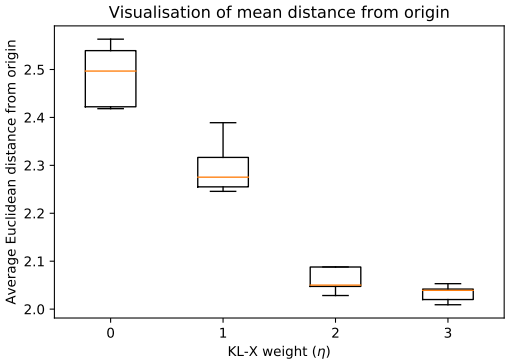


Figure 4.27: Demonstration on the effect of η using $Q = 8$ with 5 runs each on the clinical data. This is generated from the average Euclidean distance of the latent embeddings from the origin for each run.

values of η using a toy dataset (for this demonstration, we use the reduced MNIST dataset described in Section 2.12.1.1). We made use of the setting $M = 20$ and $Q = 2$ with recognition models obtained in Section 4.1.1 and performed 20 optimisation runs for each value of η.

In Figure 4.26, we visualise the latent embeddings obtained using various values of η. It can be seen that an effective value for η lies in the range of 1.5 to 5 as values in this range move the latent embeddings to be around the origin (in comparison to $\eta = 1$) and thereby offer sufficient regularisation.

We can also observe that as the value of η begins to exceed 10, the clustering structure begins to disappear as the KL term associated with X (i.e., $\mathrm{KL}_{\boldsymbol{X}}$ in Equation (4.1)) begins to dominate and the optimisation tries to move the latent points closer to zero while making them appear to be a sample from the standard normal distribution. We also studied the effect of η on the Parkinson's disease dataset. The analysis was repeated for different values of η and the number of latent dimensions was fixed to $Q = 8$ as this gave the highest predictive log-likelihood in the previous analysis. We then computed the mean Euclidean distance of the latent embeddings from the origin for each of the 5 runs. In Figure 4.27, we can observe the relative decrease in the Euclidean distance as the hyper-parameter η increases.

This analysis shows that incorporating a weight on $\mathrm{KL}_{\boldsymbol{X}}$ (i.e., $\eta > 1$) results in latent embeddings that are generally centred around the origin with results that are qualitatively consistent with Equation (3.17) (i.e., $\eta = 1$).

Chapter 5

Discussion and Conclusion

This work proposes a generative model that is targeted to clinical datasets that comprise of heterogeneous, high-dimensional data from several disparate sources. We extend the Gaussian process latent variable model (GPLVM) to produce low-dimensional embeddings of heterogeneous datasets while preserving the similarities between the observations by learning a shared latent representation. In this work, we adapt the inference framework proposed in Titsias and Lawrence [2010] and back-constrain the latent space using recognition models. We demonstrated the effectiveness of our model on toy datasets and clinical data of Parkinson's disease patients from the HUS Helsinki University Hospital. Our approach identifies sub-groups from the heterogeneous patient data and we also demonstrate the statistical significance of the findings. The differences in characteristics among the identified clusters were also evaluated using standard statistical tests. This work ties together many existing models and techniques in the field of generative modelling and demonstrates its effectiveness on clinical data.

We believe that there are many avenues for future research in generative modelling for clinical and biological data. A possible future work would involve the modelling of disease progression in our setting by recovering a 1-D manifold (pseudotime) [Trapnell et al., 2014]. It would be interesting to incorporate the ideas of semi-supervised learning as it may allow the incorporation of diagnosis (or sub-group) information which may only be available for a few patients. Moreover, active learning techniques to incorporate expert knowledge to guide the overall clustering procedure may be of significant interest to clinicians. We also seek to improve our model through better estimation of missing values as sparsity is a significant problem in clinical and biological datasets. Another avenue of future research may be the incorporation of more expressive posterior distributions in the variational inference

through the application of flow-based models.

Machine learning can play a vital role in personalised as well as precision medicine and has caused a seismic shift on how clinical patient data is being used and interpreted. Modern machine learning techniques can be harnessed for more effective and efficient healthcare that can benefit both patients and medical practitioners. Deep learning algorithms are already being exploited to try to detect and diagnose heart disorders (atrial fibrillation), predict the onset of diabetes and so much more [Topol, 2019]. Generative modelling is just one approach to this final goal of intelligent diagnostics in which the underlying distribution is modelled in order to better understand various latent properties. We hope that this work would act as a foundation to more powerful generative models for disease stratification.

References

Peter Achenbach, Katharina Warncke, Jürgen Reiter, Heike E Naserke, Alistair JK Williams, Polly J Bingley, Ezio Bonifacio, and Anette-G Ziegler. Stratification of type 1 diabetes risk on the basis of islet autoantibody characteristics. *Diabetes*, 53(2):384–392, 2004.

Emma Ahlqvist, Petter Storm, Annemari Käräjämäki, Mats Martinell, Mozhgan Dorkhan, Annelie Carlsson, Petter Vikman, Rashmi B Prasad, Dina Mansour Aly, Peter Almgren, et al. Novel subgroups of adult-onset diabetes and their association with outcomes: a data-driven cluster analysis of six variables. *The Lancet Diabetes & Endocrinology*, 6(5):361–369, 2018.

Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information retrieval*, 12(4):461–486, 2009.

Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of Marchine Learning Research*, 18:1–43, 2018.

JM Bernardo, MJ Bayarri, JO Berger, AP Dawid, D Heckerman, AFM Smith, M West, et al. The variational bayesian em algorithm for incomplete data: with application to scoring graphical model structures. *Bayesian statistics*, 7:453–464, 2003.

Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

David M Blei, Michael I Jordan, et al. Variational inference for dirichlet process mixtures. *Bayesian analysis*, 1(1):121–143, 2006.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

Thang D Bui and Richard E Turner. Stochastic variational inference for gaussian process latent variable models using back constraints. In *Black Box Learning and Inference NIPS workshop*, 2015.

Andreas C Damianou, Michalis K Titsias, and Neil D Lawrence. Variational inference for latent variables and uncertain inputs in gaussian processes. *The Journal of Machine Learning Research*, 17(1):1425–1486, 2016.

Philip J Davis and Philip Rabinowitz. *Methods of numerical integration*. Courier Corporation, 2007.

Carl Henrik Ek, Philip HS Torr, and Neil D Lawrence. Gaussian process latent variable models for human pose estimation. In *International workshop on machine learning for multimodal interaction*, pages 132–143. Springer, 2007.

Yarin Gal, Yutian Chen, and Zoubin Ghahramani. Latent gaussian processes for distribution estimation of multivariate categorical data. In *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

Samuel Gershman, Matt Hoffman, and David Blei. Nonparametric variational inference. *arXiv preprint arXiv:1206.4665*, 2012.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

Alison Harvey, Angela Brand, Stephen T Holgate, Lars V Kristiansen, Hans Lehrach, Aarno Palotie, and Barbara Prainsack. The future of technologies for personalised medicine. *New biotechnology*, 29(6):625–633, 2012.

James Hensman, Alexander Matthews, and Zoubin Ghahramani. Scalable Variational Gaussian Process Classification. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, pages 351–360. PMLR, 2015.

Ralf Herbrich, Neil D Lawrence, and Matthias Seeger. Fast sparse gaussian process methods: The informative vector machine. In *Advances in neural information processing systems*, pages 625–632, 2003.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *ICLR*, 2(5):6, 2017.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14 (1):1303–1347, 2013.

Paul Horton and Kenta Nakai. A probabilistic classification system for predicting the cellular localization sites of proteins. In *Ismb*, volume 4, pages 109–115, 1996.

Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

Tommi S Jaakkola and Michael I Jordan. Improving the mean field approximation via the use of mixture distributions. In *Learning in graphical models*, pages 163–173. Springer, 1998.

Johan Ludwig William Valdemar Jensen et al. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta mathematica*, 30:175–193, 1906.

Ian Jolliffe. Principal component analysis. In *International encyclopedia of statistical science*, pages 1094–1096. Springer, 2011.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

Nandakishore Kambhatla and Todd K Leen. Dimension reduction by local principal component analysis. *Neural computation*, 9(7):1493–1516, 1997.

Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons, 2009.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Stefan Klein, Josien PW Pluim, Marius Staring, and Max A Viergever. Adaptive stochastic gradient descent optimisation for image registration. *International journal of computer vision*, 81(3):227, 2009.

Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

Neil D Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *Advances in neural information processing systems*, pages 329–336, 2004.

Neil D Lawrence and Joaquin Quiñonero-Candela. Local distance preservation in the GP-LVM through back constraints. In *Proceedings of the 23rd international conference on Machine learning*, pages 513–520. ACM, 2006.

Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Ping Li and Songcan Chen. A review on gaussian process latent variable models. *CAAI Transactions on Intelligence Technology*, 1(4):366–376, 2016.

Qing Liu and Donald A Pierce. A note on gauss-hermite quadrature. *Biometrika*, 81(3):624–629, 1994.

David Lowe and Michael E Tipping. Neuroscale: Novel topographic feature extraction using rbf networks. In *Advances in Neural Information Processing Systems*, pages 543–549, 1997.

Pablo Moreno-Muñoz, Antonio Artés, and Mauricio Álvarez. Heterogeneous multi-output gaussian process prediction. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 6712–6721. Curran Associates, Inc., 2018. URL `http://papers.nips.cc/paper/7905-heterogeneous-multi-output-gaussian-process-prediction.pdf`.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Nikolaos Nasios and Adrian G Bors. Variational learning for gaussian mixture models. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 36(4):849–862, 2006.

John Paisley, David Blei, and Michael Jordan. Variational bayesian inference with stochastic search. *arXiv preprint arXiv:1206.6430*, 2012.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Saurabh Prasad and Lori Mann Bruce. Limitations of principal components analysis for hyperspectral target recognition. *IEEE Geoscience and Remote Sensing Letters*, 5(4):625–629, 2008.

Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

Joaquin Quiñonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research*, 6(Dec):1939–1959, 2005.

Carl Edward Rasmussen and Christopher KI Williams. *Gaussian process for machine learning*. MIT press, 2006.

Douglas Reynolds. Gaussian mixture models. *Encyclopedia of biometrics*, pages 827–832, 2015.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *International conference on artificial neural networks*, pages 583–588. Springer, 1997.

Gideon Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

Aaron Shon, Keith Grochow, Aaron Hertzmann, and Rajesh P Rao. Learning shared latent structure for image synthesis and robotic imitation. In *Advances in neural information processing systems*, pages 1233–1240, 2006.

AJ Smola, B Schölkopf, and P Langley. Sparse greedy matrix approximation for machine learning. In *17th International Conference on Machine Learning, Stanford, 2000*, pages 911–911, 2000.

Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, pages 1257–1264, 2006.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

Richard Sutton. Two problems with back propagation and other steepest descent learning procedures for networks. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society, 1986*, pages 823–832, 1986.

Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500): 2319–2323, 2000.

Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016. URL `http://arxiv.org/abs/1605.02688`.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial Intelligence and Statistics*, pages 567–574, 2009.

Michalis Titsias and Neil D Lawrence. Bayesian gaussian process latent variable model. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 844–851, 2010.

Michalis Titsias and Miguel Lázaro-Gredilla. Doubly stochastic variational bayes for non-conjugate inference. In *International Conference on Machine Learning*, pages 1971–1979, 2014.

Eric Topol. *Deep medicine: how artificial intelligence can make healthcare human again*. Hachette UK, 2019.

Cole Trapnell, Davide Cacchiarelli, Jonna Grimsby, Prapti Pokharel, Shuqiang Li, Michael Morse, Niall J Lennon, Kenneth J Livak, Tarjei S Mikkelsen, and John L Rinn. The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. *Nature biotechnology*, 32(4):381, 2014.

Naonori Ueda and Zoubin Ghahramani. Bayesian model search for mixture models based on optimizing variational bounds. *Neural Networks*, 15(10): 1223–1241, 2002.

Martin J Wainwright, Michael I Jordan, et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008.

Christopher KI Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. In *Advances in neural information processing systems*, pages 682–688, 2001.

Anqi Wu, Nicholas G Roy, Stephen Keeley, and Jonathan W Pillow. Gaussian process based nonlinear latent structure discovery in multivariate spike train data. In *Advances in Neural Information Processing Systems*, pages 3496–3505, 2017.

Eric P Xing, Michael I Jordan, and Stuart Russell. A generalized mean field algorithm for variational inference in exponential families. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pages 583–591. Morgan Kaufmann Publishers Inc., 2002.